1. Paper
    1. EEL3111 Force Sensor Paper July 2010
2. Power Point
    1. EEL3111 Force Sensor Power Point July 2010
3. Tutorial
    1. EEL3111 Student Tutorial
4. PCB
    1. EEL3111 Project PCB
5. C Code Listing
    1. EEL3111 Code Listing

EEL3111 Force Sensor Paper July 2010

This is a paper created by students in a Basic Circuit Theory course at the University of North Florida during Summer 2010. The students were tasked with using a Texas Instruments MSP430 microcontroller along with an analog sensor of their choice to create an embedded system. These students decided to use a resistive force sensor fed into a unity gain buffer and then to the ADC of the msp430. A simple program was created in C that allows the students to show a force in grams on an LCD display when pressure is applied to the sensor. This paper details the process that was followed.

Force Sensor

Juan Castillo, Mordecai L. Huber, Bernadette Quijano

College of Computing, Engineering, and Construction

University of North Florida

July 2010

ABSTRACT

This paper will discuss the different elements of our functioning force sensor. These include: the components thereof, sensor physics, accuracy of the device, and possible applications; both present and future.

Keywords
Unity Gain Buffer (UGB), Liquid Chrystal Display (LCD), Force Sensor (FSR).

Introduction

The purpose of this assignment was to build a functioning sensor using the MSP430F449 microprocessor. The sensor should generate an analog to digital signal which can then be displayed on the provided LCD. Our team has constructed a functioning force sensor. The sensor consists of four elements. The first element consists of the various power supplies used to power up the different components of the device. The second element is a voltage divider network which includes a variable resistance force sensor. The third element is an op-amp which functions as a UGB. The final element consists of the MSP430F449 microprocessor and its associated LCD. Together these four elements can detect and output the magnitude of a force applied to the sensing element (see fig. 1).

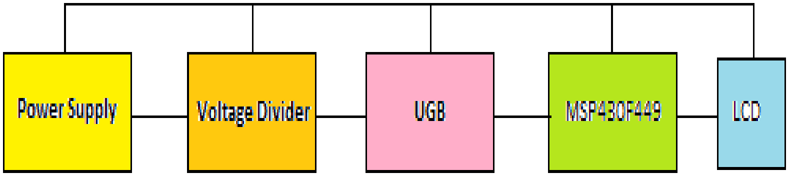


Figure 1

Power Supplies

The main power source for the device is a single nine volt battery. A simple sliding on-off switch is used to connect the device to the battery when operation is desired. The force sensing voltage divider network requires a five volt supply. A wire is run from the battery to a standard five volt voltage regulator that has been soldered to the board. A separate 3.3 volt supply is required for the MSP430F449 microprocessor and its associated LCD. This is made possible by utilizing the provided 3.3 volt precision voltage regulator that was supplied with the project kit. The 3.3 volt supply also provides the rail voltages for the UGB. This configuration ensures that the maximum signal input to the microprocessor does not exceed 3.3 volts (see fig. 2).

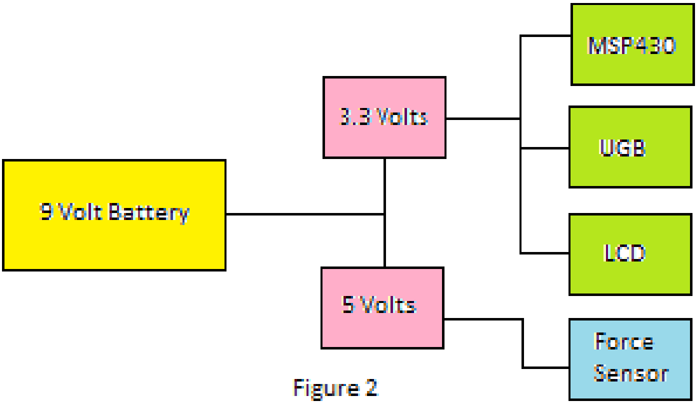


Figure 2

Voltage Divider Network

The voltage divider network consists of the above mentioned five volt supply, a variable resistance FSR, and a 10 kilo-ohm resistor. The components are wired as shown below:

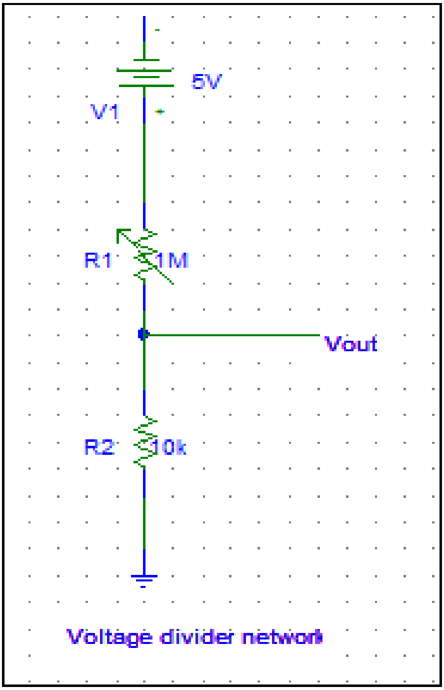


Figure 3: Voltage Divider Schematic

Force Sensor

The FSR used in this device can be purchased online at sparkfun.com. The FSR is a force sensitive resistor with a square, 1.75 x 1.5" sensing area. Its resistance will vary depending on how much pressure is being applied to the sensing area; the harder the force, the lower the resistance. When no pressure is being applied to the FSR its resistance will be larger than 1MΩ [1]. The following is a detailed explanation of the inner workings of the FSR used in our device. In a nutshell the FSR is made up of two enmeshed conductors embedded in a piezosensitive conducting film. The two conductors do not make direct contact with each other. Instead the film acts as a variable resistance barrier between them. This film is a polymer matrix composed of very small conductive elements held in suspension by a non-conducting substance [1]. The sensing film is compressible and elastic. This compressibility allows the position of the conducting elements to change when under the pressure of a load. As the load is applied some of the conducting elements butt up against the surfaces of the enmeshed conductors. In addition, the conducting particles themselves are moved closer together as a result of the downward force. These actions allow the resistance of the sensor as a whole (R_{ab}) to decrease under an applied load. Because the film is elastic the conducting film reverts back to its original position and shape when the load is removed. Because there are no semiconductors inside the sensor it is generally impervious to damage by most electromagnetic fields. It should be noted that forces capable of stressing the film or either of the conductors beyond the elastic range can cause irreversible damage to the sensor.

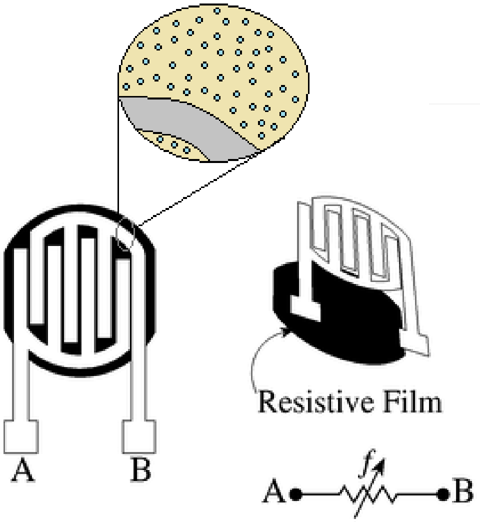


Figure 3.1: Two enmeshed conductors with sensing film

Note the term pressure which means force per unit area. While advertised as a force sensor, in reality, the device actually is pressure sensitive. Testing has shown that the same 100 gram weight will cause a greater output voltage swing if the force is concentrated on a smaller area. Based on the above description it should be clear why. Localized high pressure on a given area of the sensor creates a very low resistance bridge between the enmeshed conductors at that point. Spreading the force over a larger area will decrease the resistance of the sensor as a whole but will not result in the very low resistance circuit created under high pressure.

Operation

The 10 kilo-ohm resistor allows the circuit to act as a voltage divider. It also restricts the buildup of excessive current flow when pressure is exerted on the sensor. The equation for V_{out} is shown below:

V_{out} =

$$5V\left[\frac{10k}{10k + \mathbf{RFSR}}\right]$$

R_{FSR} represents the sensor’s resistance under a given load. In this configuration the output voltage is very small when no load is applied. The voltage increases in a nonlinear fashion with respect to the applied load. This is due to the manner in which the resistance of the FSR changes with the applied load (See figure 4). Note that only as a substantial force is applied does the rate of change of resistance with respect to applied load move towards a constant value. Switching the positions of the two resistors would cause voltage to be at a maximum under no-load conditions.

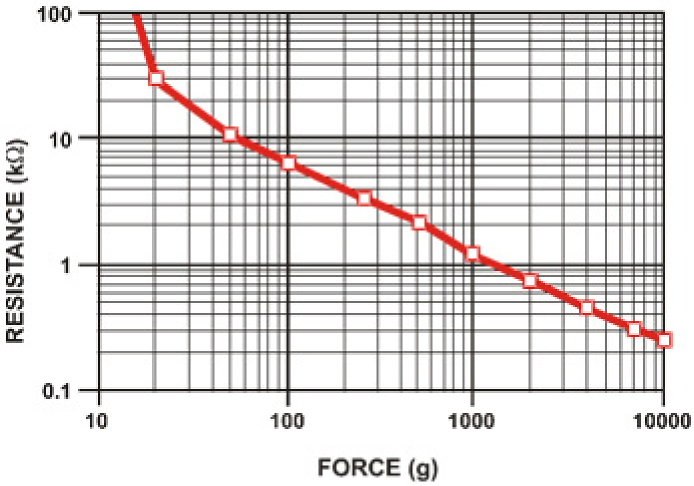


Figure 4 : Resistance varies non-linearly with applied load.

UGB

As discussed above, V_{out} is the voltage pickoff from the FSR network. This voltage is sent next to an op-amp that functions as a UGB. This is accomplished by connecting the output to the inverting input terminal [4]. No feedback resistor is used just a copper wire. This connection drives the gain of the amp towards unity and no amplification of the signal occurs. As discussed in class the purpose of this is to isolate the input to the microprocessor from the FSR circuit (fig. 5).

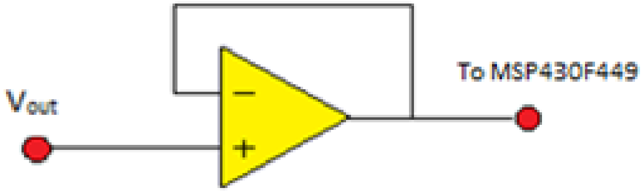


Figure 5: Unity Gain buffer

The op amp we used has the part number TLV2765. It was soldered directly on to the board. The rail voltages are 3.3 volts to the positive and ground (reference voltage) which is applied to the negative. This op amp is a rail-to rail op amp meaning that high fidelity can be achieved even when the input signal (V_{out}) is very close to one of the rail voltages [3]. This function is essential because under no-load conditions the input signal to the op-amp is very close to the reference voltage which acts as the negative rail voltage. In fact, during construction of the device we had to hook up multiple op amps before we could find one that was faithful under no load conditions. We could have dedicated a separate -3.3 volt supply to use as the negative rail voltage but selection of this op amp made that unnecessary. Full credit should be given to Mike Toth for solving this design problem.

MSP430F449 and LCD

Up until this point in the circuit the signal representing the force applied was analog. The real magic takes place inside the microprocessor where the signal is converted in to a digital signal which is suitable for display on the LCD. Again we want to give credit to Mike Toth in the lab for doing most of the programming that was required. The MSP430F449 is a microcontroller unit complete with timers, analog to digital converters, and a LCD driver. It also contains memory which can be used to program the device to perform specific functions. The primary use for such a device is to capture analog signals, convert them to digital, and output the digital signal to an LCD or to some other device [2]. They can be purchased or procured as student samples at the Texas Instrument website. Figure six is a flow chart provided by Mike Toth that illustrates the flow of the program we used in this device. Notice the infinite loop (wait forever) and the one second interrupt. The infinite loop tells the program to execute a command with no predefined condition (that is not *always* true). Technically speaking this is like telling a program to run without ever giving it instructions on when to stop. This means that the only way to halt operation is to power down the device using the sliding switch on the front of the package. In C, the programming language used by engineers, there are different ways to exit an infinite loop. You can use the break command to end the cycle when a specific condition is encountered. Another method is to include in the program what is called an interrupt.

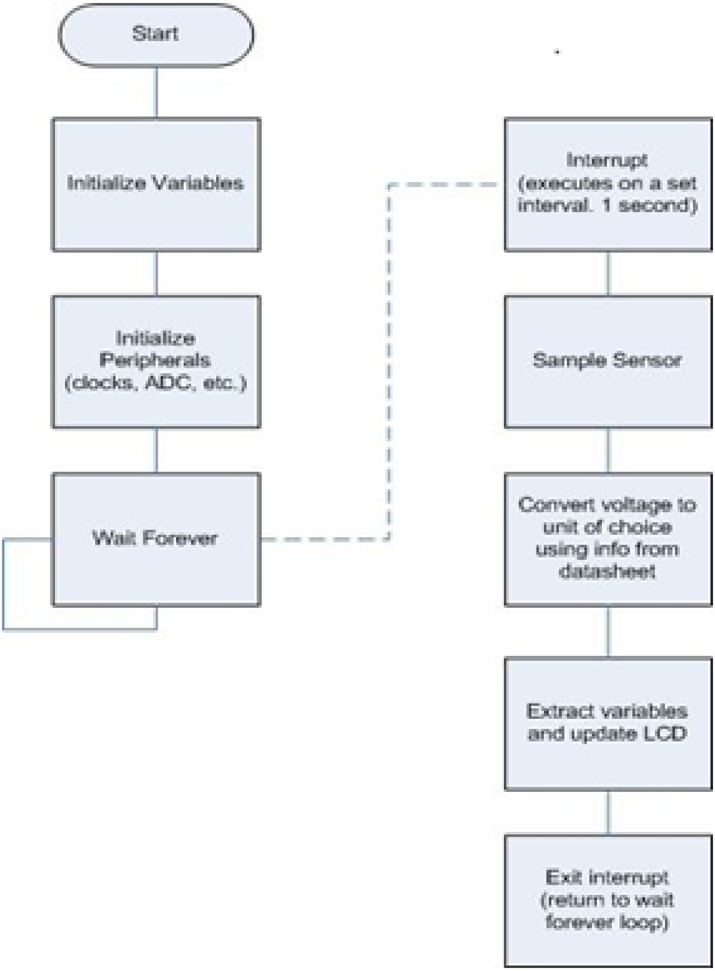

Start
Initialize Variables
Initialize Peripherals (clocks, ADC, etc.)
Wait Forever
Interrupt (executes on a set interval. 1 second)
Sample Sensor
Convert voltage to unit of choice using info from datasheet
Extract variables and update LCD
Exit interrupt (return to wait forever loop)

Figure 6: Programming flowchart

In this case, an interrupt is programmed which tells the device to sample the input from the FSR and

UGB network, convert it to digital, and update the LCD with the new information. This interrupt occurs one time per second. What this means is that the sensor is sampled and the LCD is updated only once per second. After the sampling, control is handed back to the loop until the next interrupt occurs.

The Equation

Inside the microprocessor the instantaneous value of the analog signal is converted into a digital value and stored under a previously declared floating point variable. The following equation is used to calculate the actual weight in grams of the object on the sensor. The data is displayed on the LCD.

$$y = 14.609x^4 - 38.65x^3 + 47.017x^2 + 10.847x - .2188$$

In the equation, “y” equals the weight in grams and “x” is the value stored under the floating point variable that is representative of the analog signal present on the microprocessor’s input pin. This equation was derived from the specifications sheet of the FSR when using a ten kilo-ohm resistor (See figure seven, use the purple plot). Using a different resistor might have allowed the use of a simpler equation. Bear in mind however that the rail voltage to the UGB was only 3.3 volts. This was necessary because 3.3 volts was the max allowable input to the microprocessor. The 10 kilo-ohm resistor yields a maximum output very close to 3.3 volts and was thus ideal.

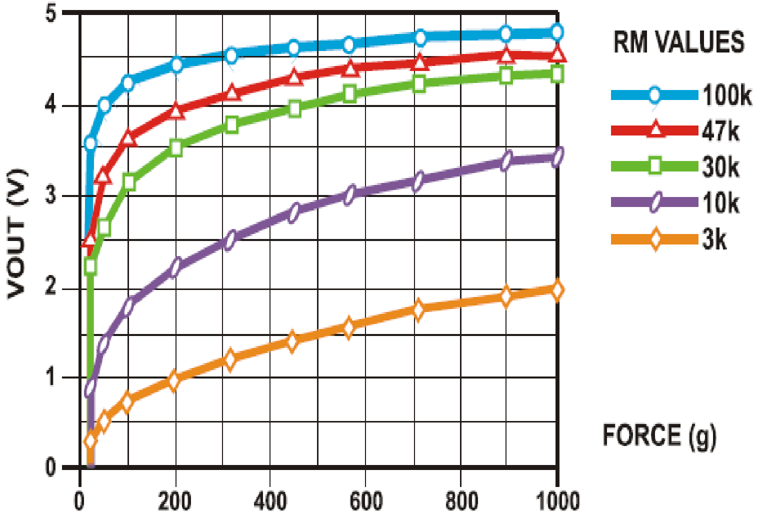


Figure 7: Graph of V(f)

Because the equation is cumbersome it takes the microprocessor some time to crunch the numbers. Mike Toth suspected that this is the cause of the flickering on the LCD. It turns out that the microprocessor we used is better suited for solving linear or quadratic equations in any efficient manner.

Accuracy of the device

What we have described is the inner workings of a device that can sense and output the magnitude of a force applied to its sensing element. The range of the device is between zero and 890 grams. Ideally the device should take measurements up to 1000 grams however the UGB becomes saturated before that is allowed to occur. The device is not terribly accurate but definitely can be used to get the sense that a force is being applied to the FSR. Some of the ways that accuracy can be improved are as follows. Firstly, we could invest in a more expensive sensor (we bought ours for $7.95). Secondly, we could do more testing to fine tune the equation used in the calculation. A piecewise continuous function might be more suitable. Finally, one of the main flaws in the device is the fact that pressure is not evenly applied across the entire sensing area. Some type of mechanism that evenly distributes the weight across the FSR is sure to boost the accuracy of the device (see figure 8).

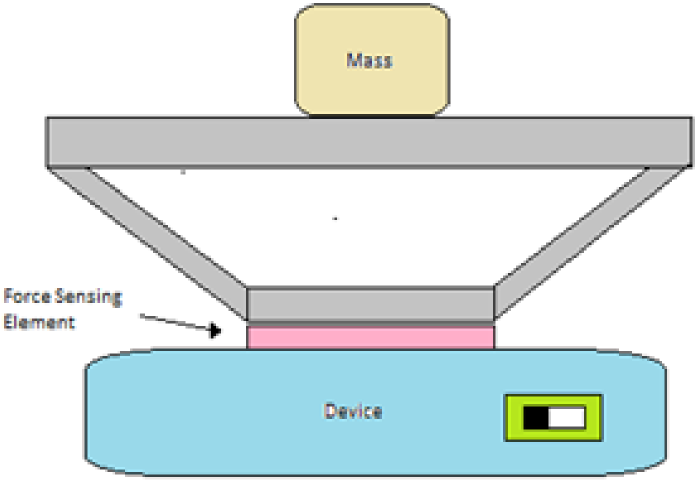


Figure 8: Mechanism to reduce pressure points.

Applications and Conclusion

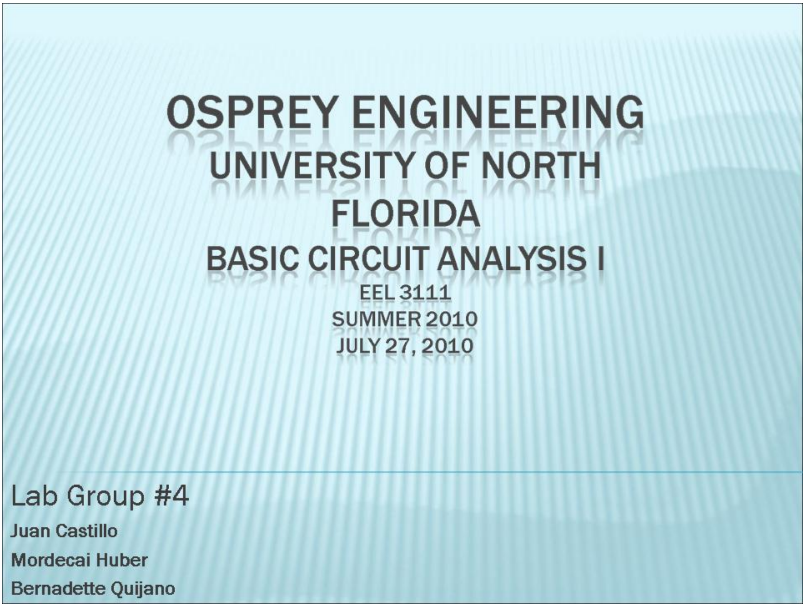
There are many different applications for force sensors that function in a manner similar to ours. The first one that comes to mind is the scale. The compact, rugged, and inexpensive nature of our sensor lends itself very well to those small digital scales used for weighing food portions. The only issue with the scale is that usually a high degree of accuracy is expected. Assuming we can correct that about our device a scale would be a perfect application for our creation. In all branches of engineering smart applications are the ways of the future. Force sensors can be a part of many different types of “smart” infrastructure. Roads and bridges are great places to incorporate force sensors. Transportation engineers could use them to gauge strain on roadways and electrical engineers could incorporate them into the design of the new red light camera systems that are coming our way soon. If they were only activated as car approaches their efficiency in terms of tickets per kilo-joule could be increased. In the near future force sensors may find their way into advanced applications such as robotics. Japan is the world leader in developing human like robots called androids. One of the more difficult tasks in mimicking human behavior is the act of walking. Most walking robots today have flat feet and lack the grace that humans have as they glide across a room. Instead they kind of lumber along. Force sensors could be a key element in developing a life-like walking android. Think about all of the nerve endings we have in our toes and feet that help us walk without any real effort. To conclude, this has been an excellent first opportunity to explore the possibilities that modern electronics has to offer. We look forward to taking it to the next level and one day implementing such technology in the field!

References

[1]http://www.sparkfun.com/datasheets/Sensors/Pressure/fsrguide.pdf
[2]http://focus.ti.com/lit/ds/symlink/

msp430f449.pdf
[3]http://www.swarthmore.edu/NatSci/echeeve1/Ref/SingleSupply/SingleSupply.html
[4]http://www.facstaff.bucknell.edu/mastascu/elessonshtml/OpAmps/OpAmp3Note1Buffer.html

EEL3111 Force Sensor Power Point July 2010
This is a Power Point presentation created by students in a Basic Circuit Theory course at the University of North Florida during Summer 2010. The students were tasked with using a Texas Instruments MSP430 microcontroller along with an analog sensor of their choice to create an embedded system. These students decided to use a resistive force sensor fed into a unity gain buffer and then to the ADC of the msp430. A simple program was created in C that allows the students to show a force in grams on an LCD display when pressure is applied to the sensor.



OSPREY ENGINEERING
UNIVERSITY OF NORTH FLORIDA
BASIC CIRCUIT ANALYSIS I
EEL 3111
SUMMER 2010
JULY 27, 2010

Lab Group #4
Juan Castillo
Mordecai Huber
Bernadette Quijano

OBJECTIVES

- Describe:
  - Physics of the Force Sensing Element
  - Elements of the Force Sensing Device
  - Programming of the Microprocessor
  - Accuracy of the Device
  - Applications and Conclusions

PHYSICS OF THE FORCE SENSOR

- 1.75 x 1.5 in. of area
- Two enmeshed conductors
- Piezosensitive conducting film
- Compressible
- Elastic

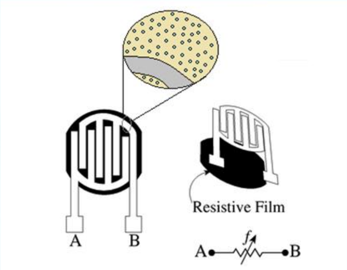
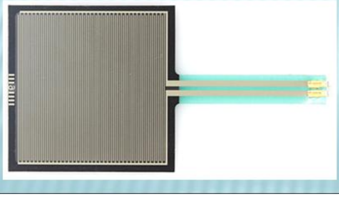


Figure 3.1: Two enmeshed conductors with sensing film



PHYSICS OF THE FORCE SENSOR

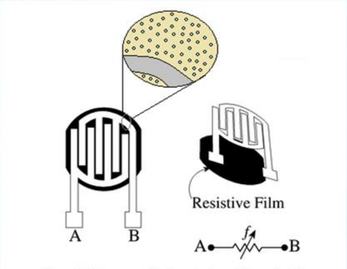
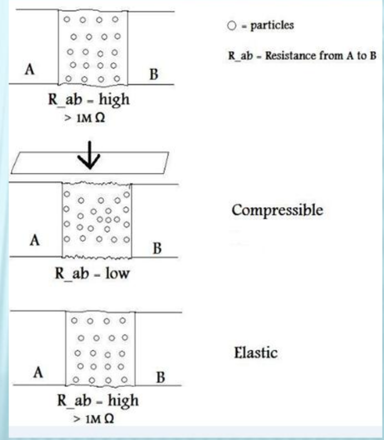
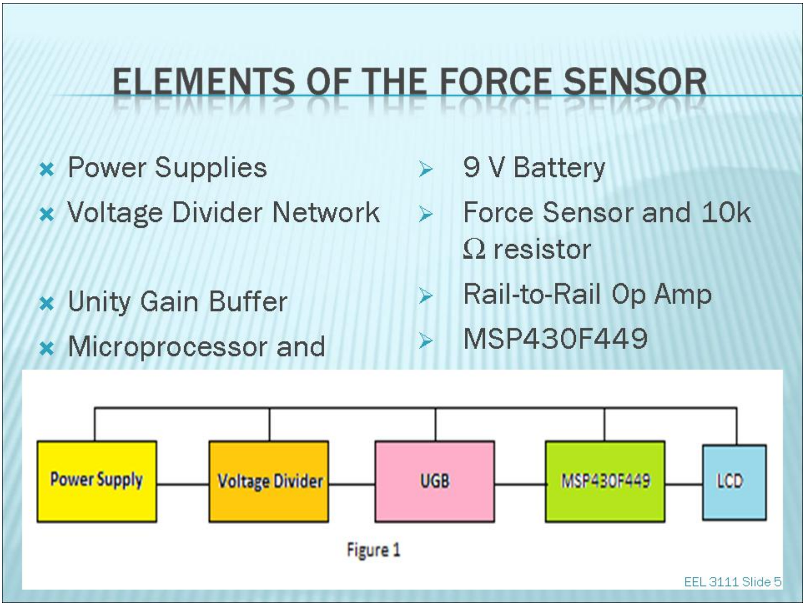


Figure 3.1: Two enmeshed conductors with sensing film



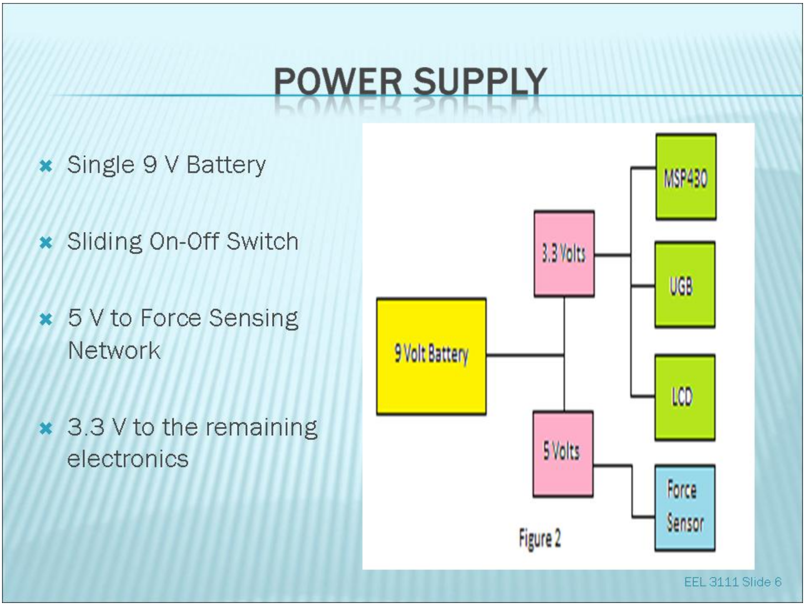


ELEMENTS OF THE FORCE SENSOR

Power Supplies	9 V Battery
Voltage Divider Network	Force Sensor and 10k Ω resistor
Unity Gain Buffer	Rail-to-Rail Op Amp
Microprocessor and	MSP430F449

Power Supply
Voltage Divider
UGB
MSP430F449
LCD

Figure 1



POWER SUPPLY

- Single 9 V Battery
- Sliding On-Off Switch
- 5 V to Force Sensing Network
- 3.3 V to the remaining electronics

Figure 2

VOLTAGE DIVIDER NETWORK

$$V_{out} = 5V\left[\frac{10k}{10k + \mathrm{RFSR}}\right]$$

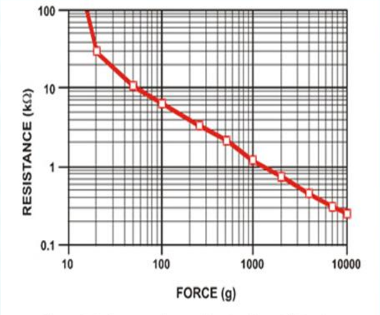


Figure 4 : Resistance varies non-linearly with applied load.

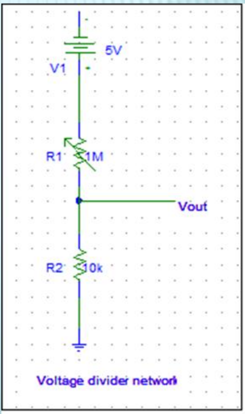


Figure 3: Voltage Divider Schematic

UNITY GAIN BUFFER

- Rail Voltages

  3.3 V and 0 V

- Gain is 1

  non inverting

- Rail-to-Rail Op Amp

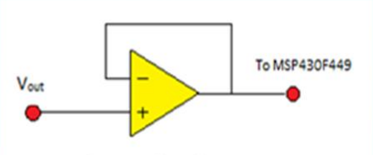


Figure 5: Unity Gain buffer

MICROPROCESSOR AND LCD

- Infinite loop

  Must turn off using switch

- Takes force measurement once per second

- 4 Digit LCD Display

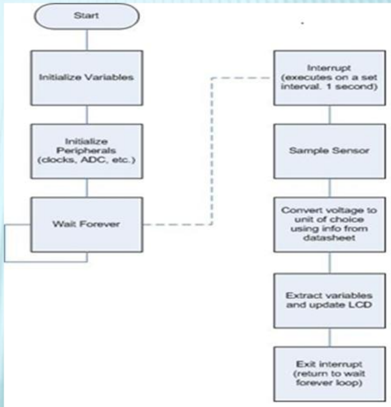

Start
Initialize Variables
Initialize Peripherals (clocks, ADC, etc.)
Wait Forever
Interrupt (executes on a set interval. 1 second)
Sample Sensor
Convert voltage to unit of choice using info from datasheet
Extract variables and update LCD
Exit interrupt (return to wait forever loop)

Figure 6: Programming flowchart

ACCURACY OF THE DEVICE

$$y = 14.609x^4 - 38.65x^3 + 47.017x^2 + 10.847x - .2188$$

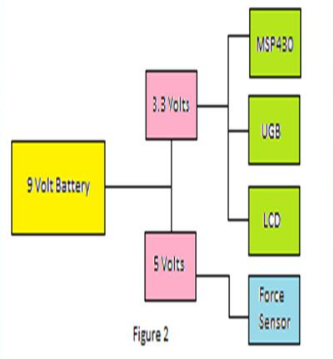


Figure 2

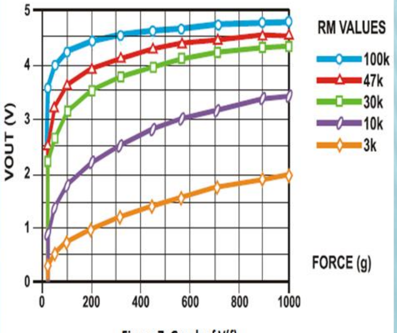


Figure 7: Graph of V(f)

IMPROVEMENTS ON THE DEVICE

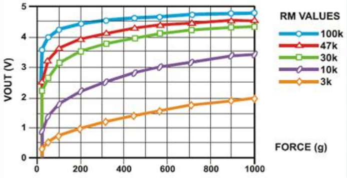


Figure 7: Graph of V(f)

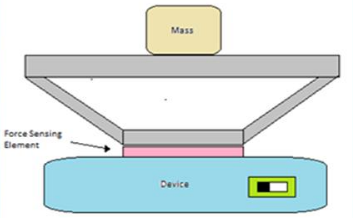
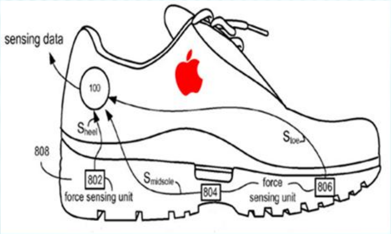
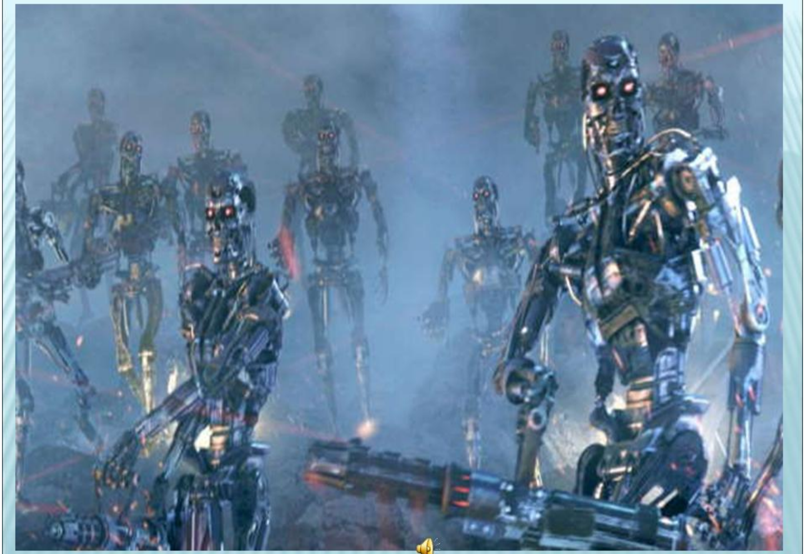


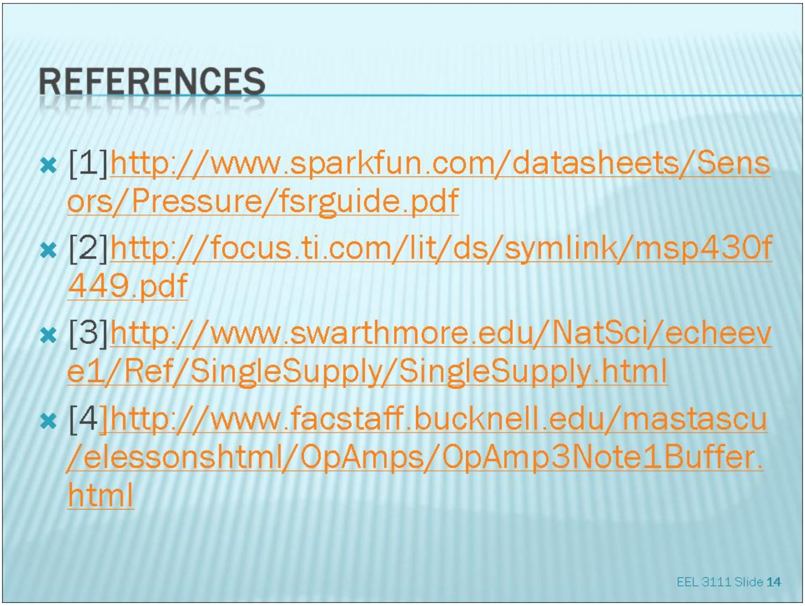
Figure 8: Mechanism to reduce pressure points.

APPLICATION AND CONCLUSION

- Measurements and Maintenance
- Future Applications







REFERENCES

- [1]http://www.sparkfun.com/datasheets/Sensors/Pressure/fsrguide.pdf
- [2]http://focus.ti.com/lit/ds/symlink/msp430f449.pdf
- [3]http://www.swarthmore.edu/NatSci/echeeve1/Ref/SingleSupply/SingleSupply.html
- [4]http://www.facstaff.bucknell.edu/mastascu/elessonshtml/OpAmps/OpAmp3Note1Buffer.html

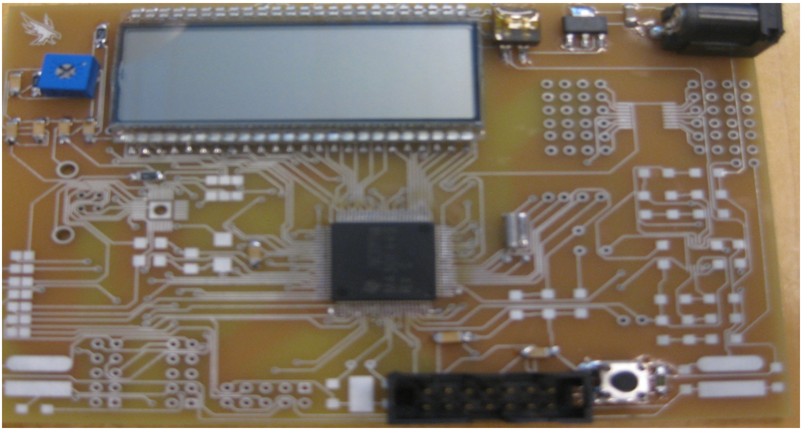
EEL3111 Student Tutorial
This is a short tutorial created by a student at the University of North Florida to help familiarize students with constructing the PCB and Flashing a program onto the microcontroller using IAR Embedded Workbench.

Electrical Engineering

University of North Florida

Embedded System Design

TI MSP430F449 Microcontroller



Mike Toth and Rasem Mourad

August 2010

Table of Contents

Topic:

- Background
- Introduction
- Board Construction
- Soldering
- Soldering Surface Mounted Components
- Soldering the MSP430
- Mounting the LCD Display
- Order for Soldering Components to Board
- Sample Devices From Industry
- Texas Instruments' MSP430F449 Samples
- Sensors
- Project Software
- Programming the MSP430F449
- Appendix

Background:

The purpose of the student tutorial is to introduce and familiarize the user with the Embedded System

Design utilizing the TI MSP430F449 microcontroller. In addition, the student tutorial serves as a guide on how to assemble the Printed Circuit Board (PCB) along with a sensor of choice. The guide also includes an introduction on programming the TI MSP430F449 using the Embedded IAR Workbench. The student will need the Flash Emulation Tool (FET) kit purchased from Texas Instruments that allows connection from the Personal Computer (PC), where the software or micro program is developed, to the PCB the student constructs. Basically, the student downloads or Flashes the micro program to the MSP430F449 Flash nonvolatile memory. More information about the TI MSP430F449 microprocessor is available at http://www.ti.com. Under Products, click on Analog and Mixed Signal MSP430 Ultra-Low-Power MCU.

The parts kit allows the student to build a circuit board with the TI MSP430F449 microprocessor, a sensor of choice, along with an LCD display. The micro program to for the TI MSP430F449 is mostly developed in the C programming language with the aid of a software package, the Embedded IAR Workbench.

After the PCB is constructed, a micro program with a message such as HELLO to be displayed on the LCD is prepared and placed in Flash memory, (flashed). A valid message on the LCD indicates that the soldering was correct and the circuits are functioning. After some familiarity is developed with the system, the student is responsible for connecting a sensor to the PCB and writing the software (micro program) that allows the sensor to display a reading on the LCD.

This student tutorial is intended for the student to have fun and to learn how to use a microprocessor to design an Intelligent Sensor. This serves as a foundation to build upon for later courses and to stimulate an interest in the art of Embedded System Design.

Introduction:

The purpose of this student project is multi-fold.

- It provides the opportunity to work as a member of a team.
- The student starts by learning to interpret a printed circuit board (PCB) layout. Several components will be provided to mount (solder) on to this board.
- Learn some basic skills of making ultra-find solder connections.
- Learn some basic steps of troubleshooting a printed circuit board (PCB).
- Pull-together files to form a project and compile and link the programs of the project.
- Introduction to the IAR debugging tool.

- Learn to flash software to the microprocessor mounted on the PCB.
- Select a sensor and learn the characteristics of the sensor.
- Write a program to allow the student to read the analog voltage of a sensor into the MSP430F449 and display the signal in appropriate units on the PCB’s LCD.
- The expected outcome from this tutorial is that the student becomes familiar, at a first level, on how several aspects of engineering (hardware construction, software development, sensors) are brought together to form an intelligent sensor.

Board Construction:

Table 1 shows a list of parts that are needed to carry out the main objectives in the student tutorial. Be sure to take an inventory of the items listed in Table 1.

There are soldering stations and microscopes available to accomplish the student tutorial. A microscope is necessary to see how to do the soldering work. This will be realized as practice in soldering is done often and over time. Solder and flux are provided with the soldering stations.

The student is expected to solder two surface mounted 0.1 uF capacitors to the printed circuit board. In addition, the user is expected to solder all 100 pins of the MSP 430F449 package to the printed circuit board.

Item #	Description (Digikey)	Quantity
1	MHC14K-ND: JTAG Connector	1
2	EG1868CT-ND: Push Button Switch	1
3	CP-102A-ND: Power Connector	1
4	LM2937IMP-3.3CT ND: Voltage Regulator	
5	CT2192MST-ND: Switch 2 position	1
6	300-1001-ND: 32.7 kHz Crystal	1
7	399-1249-1-ND: 0.1uF SMD capacitor	3
8	399-1296-1–ND: 1uF SMD	1

	capacitor		
9	311-47kECT-ND: 100 kohm resistor	2	
10	SBLCDA2: LCD Display	1	
11	circuit board - no mask - practice soldering	1	
12	circuit board - with mask - final product	1	
13	miscellaneous 24 gauge wire	2	
14	TI MSP430F449IPZ chip	1	

Inventory of Parts

Soldering:

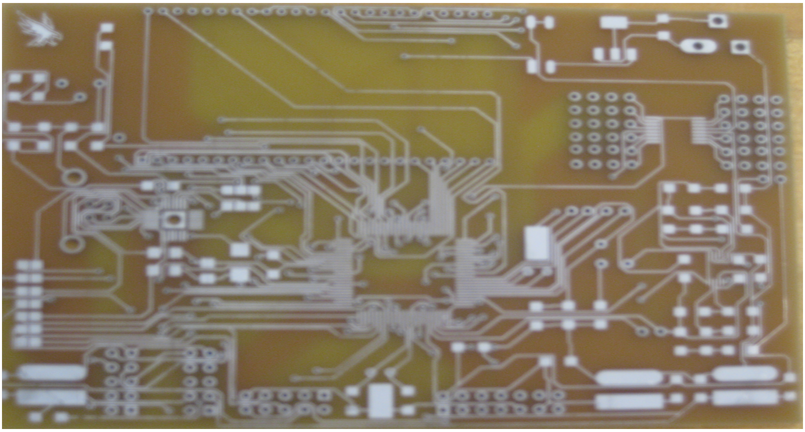
Avoid breathing the fumes from the solder. The solder is rosin core with a certain percentage of lead (about 2%). It is not healthy to breathe these fumes. Read the label on the soldering tube provided with the soldering station and use the precautions given on the label.

Set the temperature of the soldering station to 800

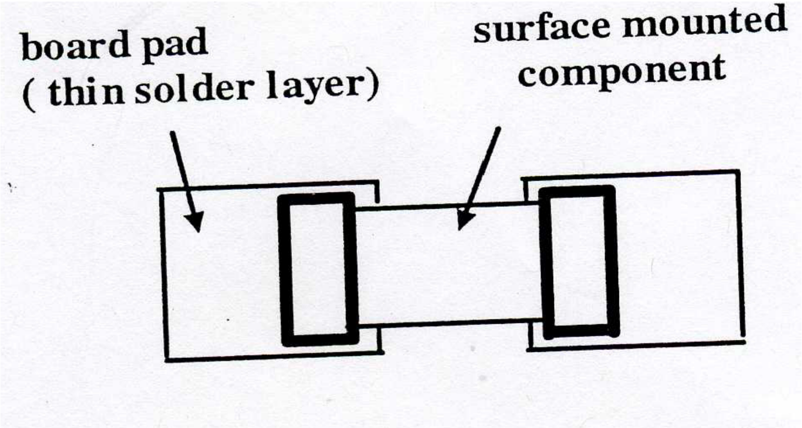
F. After the tip has reached this temperature, dip the tip of the iron into the tinning paste. Clean the tip with the sponge that comes with the station (add water to the sponge). After this step you may want to take a knife blade and lightly scrape the tip of the iron. Repeat dipping the tip in the tinning past and cleaning the tip on the sponge until the lower 1/4 inch of the tip is silvered. A clean silvered tip is the first indication that the tool has passed the first necessary step toward being properly tinned. Next, touch a strand of solder to the tip. If the solder balls up and drops from the tip, then the tip is not properly tinned. Repeat the above steps until the solder will stick to the tip. At that point the tip is properly tinned and ready for use. Another thing; when you are through soldering, always clean the tip before turning off the station. You might want to take the edge of a knife blade and lightly scrape the tip. Be careful when using the soldering iron. Soldering iron burns hurt! So does melted solder that splashes on you!
Illustration of the MSP430F449 board.Illustrating surface mounted component on a board.Soldered in place MSP430F449.Drawing showing surface mounted polarized capacitor.

Soldering Surface Mounted Components:

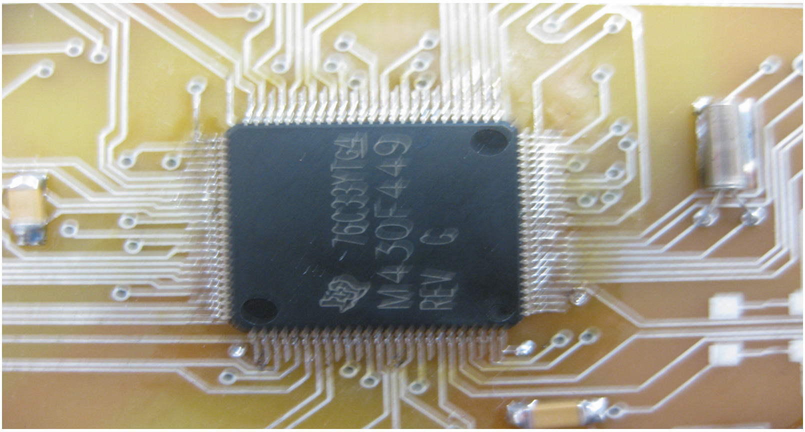
Figure 1 illustrates a top view representation of the PCB utilized to mount the MSP430F449 and the SoftBaugh LCD among other components.



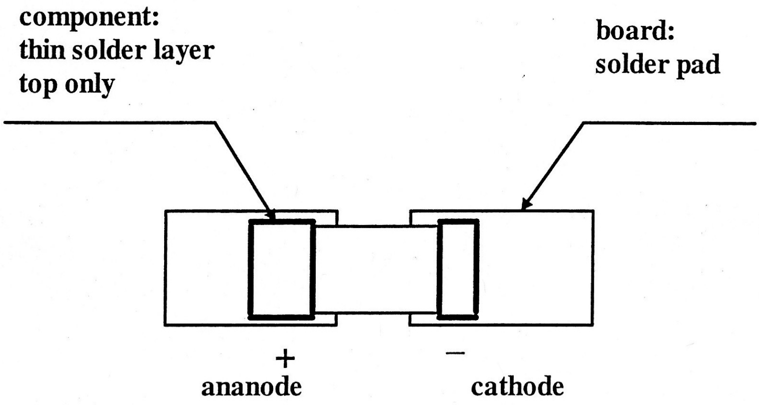
Now consider soldering a surface mounted (SMD) device such as the MSP430F449, a resistor or capacitor to a board. The PCB has a solder pad or tab for each end of the SMD pads to rest. This is indicated in Figure 2.



The PCB pads are thin and most often do not provide enough solder for connecting the SMD component. Before placing the SMD component on the board, one should flux and lightly tint each tab. To do this, take a toothpick or similar object and deposit and ever so slight amount of flux on each pad. Next, place the tip of the soldering iron in the tinning paste, remove and clean on the sponge. Now touch the tip to a strand of solder and tin-up the tip of the iron. Focus in on the tabs and bring the tip of the iron over one of the tabs. Touch the tip of the iron to the tab and slight rub the tip over the tab to deposit a thin layer of solder that should bond with the board tab. Do this to each tab. Now take a pair of tweezers and hold the SM component in place. Bring the tip of the iron to one edge of the tab and melt the solder. Figure 3 illustrates the tab for each pin of the MSP430F449 soldered onto the board.



Do not touch the tip to the SMD component. As the solder on the tab melts, the solder on the tip of the SMD component will also melt. In order to observe all this, a microscope will be needed. Remove the tip of the iron but hold the component in place, momentarily, for the solder to cool and set. Remove the tweezers and repeat the process at the other end of the component. Most often there is a good bond and no further soldering is required. Furthermore, there is a slight change in the procedure for mounting polarized capacitors. This slight change is illustrated in Figure 4.



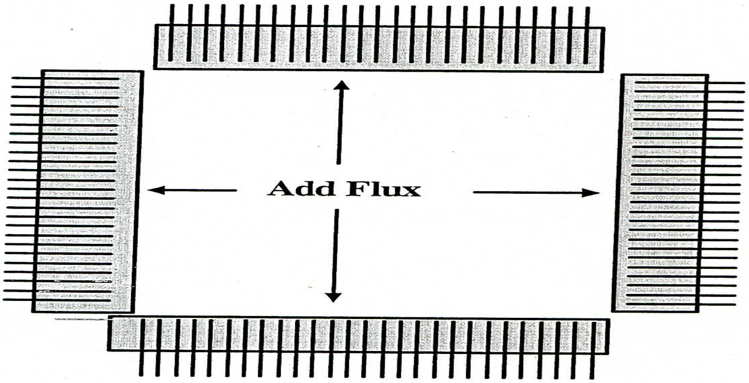
Also, on many polarized capacitors the two small solder tabs have the same size. In this case the backside of the capacitor will have a small stripe to indicate the + (positive) terminal of the capacitor. One should always check the source catalog for this information if it is not clear which side is positive.

Board pin layout for the MSP430.Showing the MSP430 placed on circuit traces.Soldering technique for connecting MSP430 to traces on board.

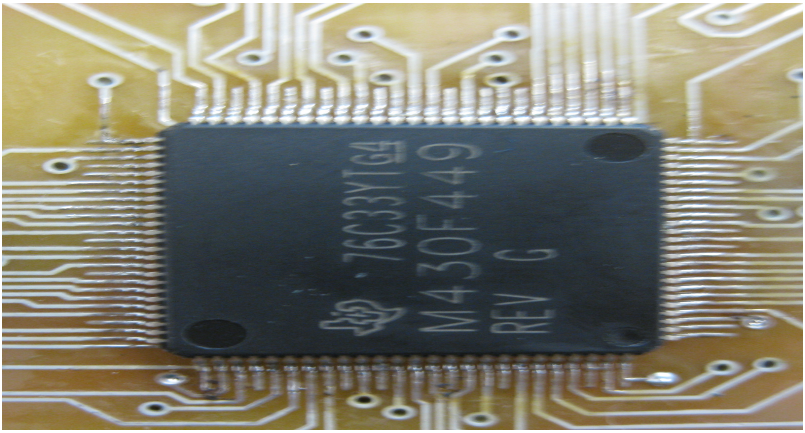
Soldering the MSP430 to the Board:

The MSP 430F449 is a very fine-pitch chip. That is, the connect point to the chip are numerous and very close together. There are 25 connect points on each side of the chip. The chip is roughly 1/2 inch on each side. That is, one is looking at roughly 25 connect points spread over 1/2 of an inch. To mount the MSP 430 to the PCB requires some tedious soldering work.

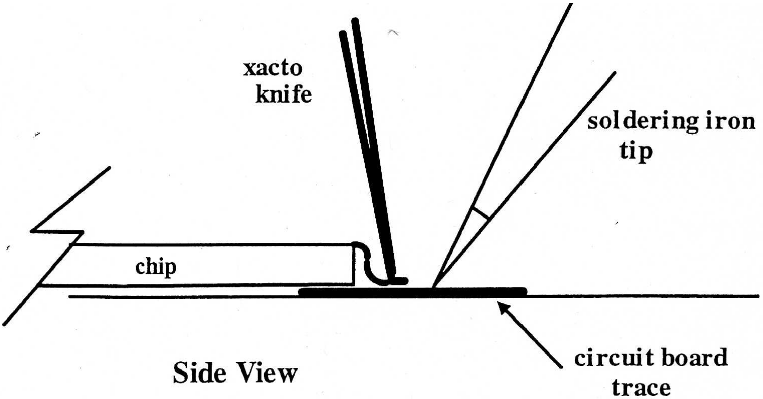
Start by placing the board on the microscope with the base light and top light of the microscope turned on. This provides heat to the board that will be helpful in the fluxing process. Consider the diagram of Figure 3 that represents the portion of the PCB where the MSP 430 will be mounted.



The student should take an appropriate object such as a toothpick with flux on the tip and place the flux in the shaded areas shown on the circuit traces of Figure 5. Next, place the MSP 430 on the trace pattern above with the MSP 430 centered all-around. This takes a little time. Be careful to align the connect points to the PCB completely around the MSP 430. The notch on the bottom left corner of the MSP 430F indicates the placement of the first pin.



With all the connect points of the MSP 430 on top of the corresponding PCB pads, hold things in place and use the soldering iron to tack a connect point of the chip to a trace at pins 25 and 1 as shown above. DO NOT ADD ANY SOLDER TO THE TIP OF THE SOLDERING IRON! Set things up as shown in Figure 5. Apply a touch of flux to the tip of each microchip pin.



At pin 25, shown in Figure 6, continue to hold the chip in place. Place the tip of the soldering iron on the circuit trace as shown above. The solder on the trace will melt. This should temporarily tack the circuit connect point to the trace. Allow the solder joint to cool for about 5 seconds. Continue to hold the chip in place. Move to another corner point on the chip (pin 1) and repeat the above process. At this point the chip should be tacked to the traces and lined up properly. Now move to another connect point (pin 24) and place the tip of an exacto knife on top of the chip connect point as shown in Figure 7. Apply the tip of the soldering iron to the corresponding circuit trace and allowing the solder on the trace to flow. Place the tip of the xacto knife on top of the connect point to the board; remove the soldering iron tip and apply a small pressure (with the xacto knife) to hold the connect point down; allow the junction to cool for a few seconds.

Then, the above process is continued by moving around the board. A good idea is to pin down connect pin 2 then move to pin 23. This allows connect point at pin 2 to cool for a few seconds before coming back to solder at pin 23 and so on. If at some point there does not seem to be enough solder on the traces to bond to the connect point, try placing the soldering iron tip in the tinning paste. Clean on the sponge. There is usually a slight amount of solder left on the tip. This can be used to add a little solder to the trace.

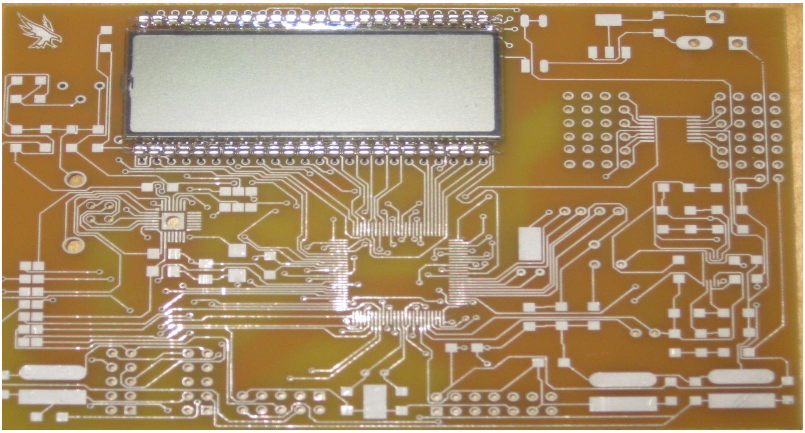
After soldering all the pins to the board, the user should check the connections. A good way to do this is to place the tip of your xacto blade to the edge of each pin. Apply a slight amount of pressure to the pin. If the pin moves you know you have a faulty solder connection. Clean and flux the tip of the soldering iron two or three times and then proceed to Re-do the solder joint.
Mounted SoftBaugh LCD Display.PCB with Components Mounted.

Mounting the LCD Display:

The SoftBaugh LCD display unit has 24 pins on each side. The following procedure establishes one manner in which to mount the unit to the board.

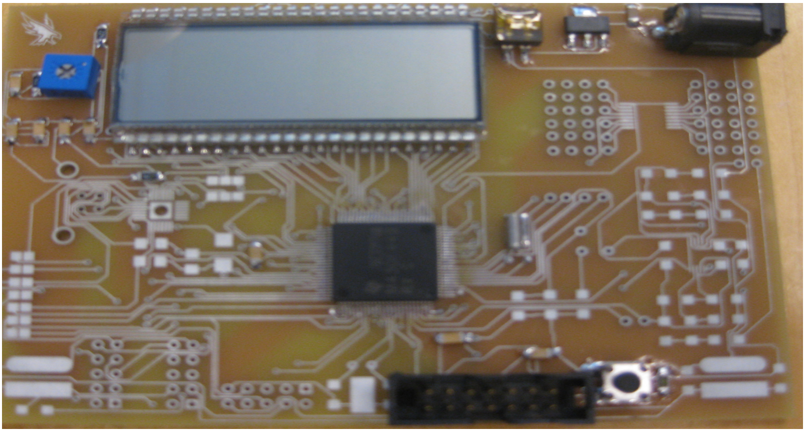
To begin with, the small notch at one end of the LCD should be on the left side when lining up pins on the board. Place the pins on one side of the unit over the corresponding holes on the board. Slightly jiggle the unit until you can get the one side to slide into the holes. There are times when slightly jiggling the unit will result in both sides of the LCD to smoothly slide into the holes of the board. However if this is not the case, do not try to force the pins into the holes. It is fairly easy to get one side of the unit to go into the board. Where the pins emerge from the bottom of the board, keep the tips about flush with the board. Then attempt to line things up with the other side of the LCD. Start at one end and use the tip of a xacto knife to move each pin over the corresponding hole. Continue to move down the pin line and line up the pins with the holes. Once the pins have lined up, the second side will fit in place. Push the LCD in place with the pins protruding from the back of the board.

Now that the LCD is in place, flip the PCB over and solder the pins to the PCB. To do this bring the solder tip next to a pin; touch the pin with the strand of solder. The solder sucks down into the hole on the PCB. Figure 8 illustrates the SoftBaugh LCD mounted on the board.



The remaining components are relatively easy to mount to the PCB. In mounting the switches, it is a good idea to apply a small amount of flux to the solder pads and even tin the tips of the switches with a slight amount of solder. This makes it easier to solder the switches to the board.

The PCB for this project was designed to serve the needs of several projects at the university. Therefore, the board is populated according to the particular project. The lettering on the board identifies the surface mounted capacitors for this tutorial and Figure 9 illustrates the board with most of the essential components mounted.



Order for Soldering Components to Board:

The following order is recommended for soldering components to the board. (1) The MSP430F449. (2) Surface mounted capacitors and resistors next. (3) Push button switch, voltage regulator, 5 volt input plus, slider switch. (4) Now solder in the pins of the JTAG connector. (5) Wait until last to solder in the SoftBaugh LCD.

Sample Devices from Industry:

Most all industries support students in their quest for learning about particular products from their company. Within reason, companies will provide samples of devices for students to use in their projects. There are at least two reasons that come to mind as to why they want to provide this support. First, they support education. Engineers, with good educations, are the foundation or backbone of their company. Second, if students become familiar with their products they are more likely to place orders with them during your professional career.

TI MSP430F449 Samples:

This is the micro computer that is utilized in this tutorial. Samples will be provided to students enrolled in the course. The following Texas Instrument URL is where you can order a 3-lot sample of the MSP430F449. Just follow the instructions. Be sure to give a mailing address where UPS/FedEx or order delivery services can deliver a package to you. You cannot use P.0.Boxes. In filling out the request form for parts indicate as necessary that you are a student working on an embedded microcomputer project.

http://focus.ti.com/docs/prod/folders/print/msp430f449.html#samples

Sensors:

After you have constructed your board and you know that you can flash a message to the LCD, the next part is for you to select a sensor for application with the PCB you have built. In particular, you are to connect the output voltage from a sensor you select to an input port to the Analog to Digital converter of the MSP 430. You might select a temperature sensor, a force sensor, a magnetic field measurement sensor and so on. The choice of sensor is left up to you. However, you must find a sensor you want to use and you must order free samples.

There are a number of companies that manufacture sensors. One company is Analog Devices. Their web address is http://www.analog.com . Other companies such as Texas Instruments and Freescale also provide sample sensors.

Project Software:

The software required for this tutorial is available on the TI website to download for free. The web address to obtain the software is:

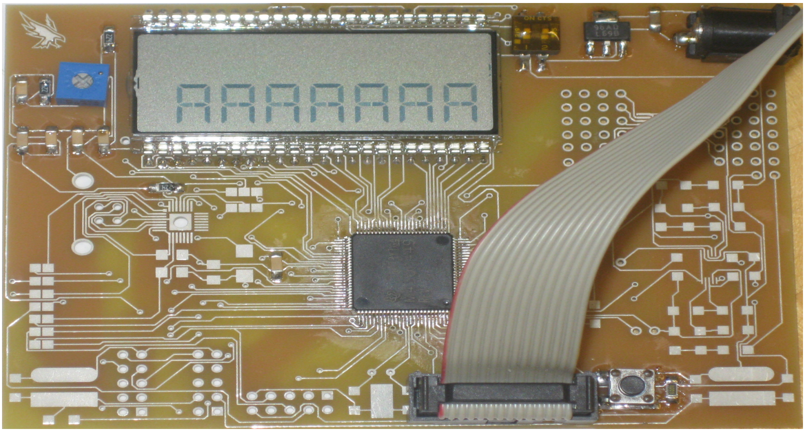
http://focus.ti.com/docs/toolsw/folders/print/iar-kickstart.html

You will find the software from both IAR and TI that supports the Flash Emulation Tool. You are encouraged to spend some time looking over this material.
Board with 14-pin connector connected to JTAG port. Showing procedure in building software, Step 2.Showing procedure in building software, Step 3.Showing procedure in building software, Step 4.Showing procedure in building software, Step 5.Showing procedure in building software, Step 6.Showing procedure in building software, Step 7.Showing procedure in building software, Step 8.Showing procedure in building software, Step 9.Showing procedure in building software, Step 10.Showing procedure in building software, Step 11.

Programming the MSP430F449:

We now explain how to use the above tool to exercise the embedded processor board so that that the LCD displays a count from A-Z.

First connect the 25-pin cable connecter of the FET to the printer port of your PC. Connect the 14-pin connector to the JTAG port on your board. Be sure this cable is connected in the Up position. This is illustrated in Figure 10.



Step 1: On your PC create the following directory: C:\430project\software. The files required to implement this student tutorial for the MSP430F449 will be given by the instructor. Once given, copy the files to your Computer. **These files may include: lcd.c, lcd.h, lcddemo.c.**

Step 2: Take the following path on your PC: **Start / Programs / IAR Systems/ IAR Embedded Workbench for MSP430 KickStart/IAR / Embedded Workbench**. This will open the screen shown in Figure 11.

Step 3: Now click on **Project/Create New Project…** and you will see the screen shown in Figure 12.

Step 4: Now double click on the programming language you will be using for the project and click on main. This step is shown in Figure 13. A Save As dialog box appears. Choose a name to save your project under.

Step 5: Once you've created your project, this will produce the screen shown in Figure 14.

Step 6: Now click on **Project/Options…** Under category, highlight general options and select which device you will be using for the project. The screen shown in Figure 15 illustrates this step.

Step 7: From the screen in Figure 15, under category highlight **Debugger** and choose the **FET Debugger** as your driver. This will produce the screen shown in Figure 16.

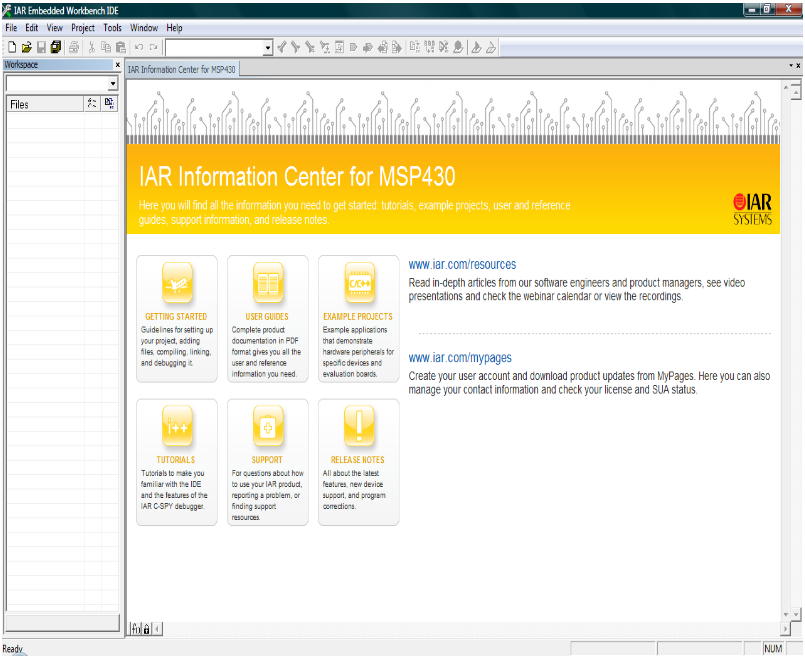
Step 8: Now under category highlight **FET Debugger**. Choose your connection whether it is USB or not. This is shown in the screen in Figure 17.

Step 9: To add the project files, click on **Project/ Add Files**. Highlight all the files needed for the project and click **Open**. The files will then be added to your workspace. This is illustrated in the screen shown in Figure 18.

Step 10: Once the files are downloaded, click on the arrow on the top right screen of your window which means you are ready to download and debug your project. This is shown in Figure 19.

Step 11: After building your project, the debugger window opens at the bottom part of your screen. Click on the three arrows encircled in Figure 20 to run the program.

Step 12: Provided that the FET cable is inserted to your PC printer port and the JTAG plug on your processor board and provided you have built your board correctly, you will see the screen go through a stage of showing that the board is being erased and then show it being programmed. After the programming stage is complete you should see some a count from A-Z scroll across the LCD.



IAR Information Center for MSP430

Here you will find all the information you need to get started: tutorials, example projects, user and reference guides, support information, and release notes.

IAR SYSTEMS

GETTING STARTED
Guidelines for setting up your project, adding files, compiling, linking, and debugging it.

USER GUIDES
Complete product documentation in PDF format gives you all the user and reference information you need.

EXAMPLE PROJECTS
Example applications that demonstrate hardware peripherals for specific devices and evaluation boards.

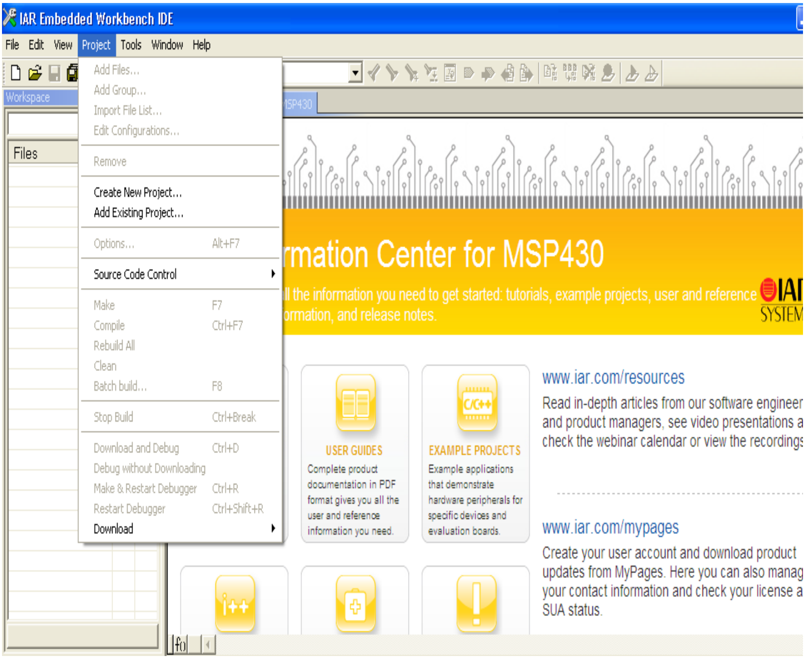
TUTORIALS
Tutorials to make you familiar with the IDE and the features of the IAR C-SPY debugger.

SUPPORT
For questions about how to use your IAR product, reporting a problem, or finding support resources.

RELEASE NOTES
All about the latest features, new device support, and program corrections.

www.iar.com/resources
Read in-depth articles from our software engineers and product managers, see video presentations and check the webinar calendar or view the recordings.

www.iar.com/mypages
Create your user account and download product updates from MyPages. Here you can also manage your contact information and check your license and SUA status.



rmation Center for MSP430

ll the information you need to get started: tutorials, example projects, user and reference ormation, and release notes.

Complete product documentation in PDF format gives you all the user and reference information you need.

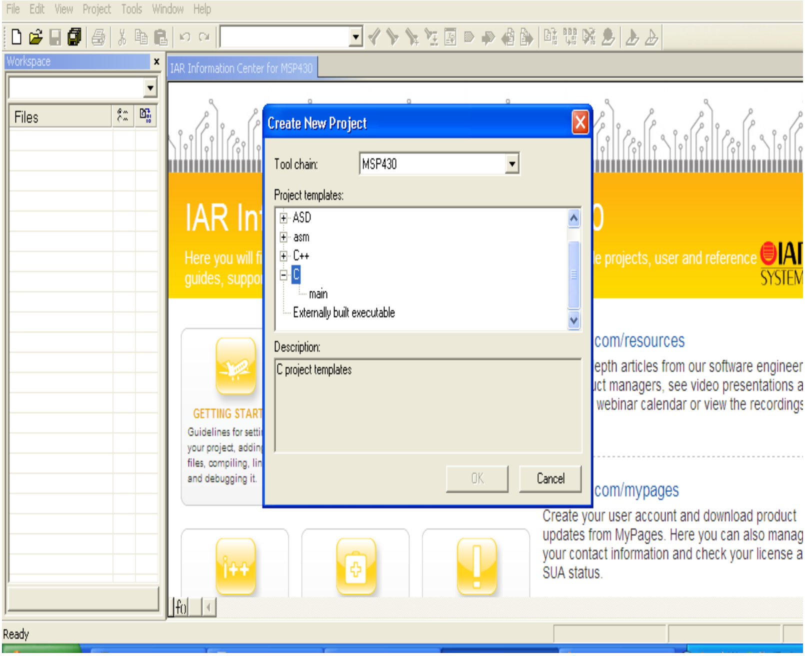
Example applications that demonstrate hardware peripherals for specific devices and evaluation boards.

www.iar.com/resources

Read in-depth articles from our software engineer and product managers, see video presentations a check the webinar calendar or view the recordings

www.iar.com/mypages

Create your user account and download product updates from MyPages. Here you can also manag your contact information and check your license a SUA status.


File Edit View Project Tools Window Help
Workspace
Files
IAR Information Center for MSP430
OK
Cancel
Ready

Create New Project

Tool chain: MSP430

Project templates:

- ASD
- asm
- C++
- C
  - main
- Externally built executable

Description:

C project templates

IAR In

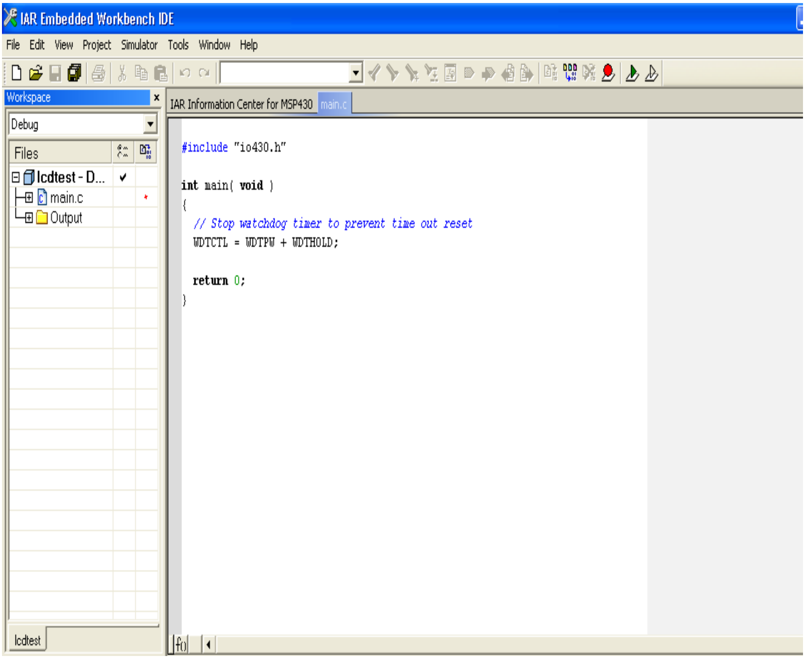
Here you will f ... e projects, user and reference
guides, suppo

IAR SYSTEM

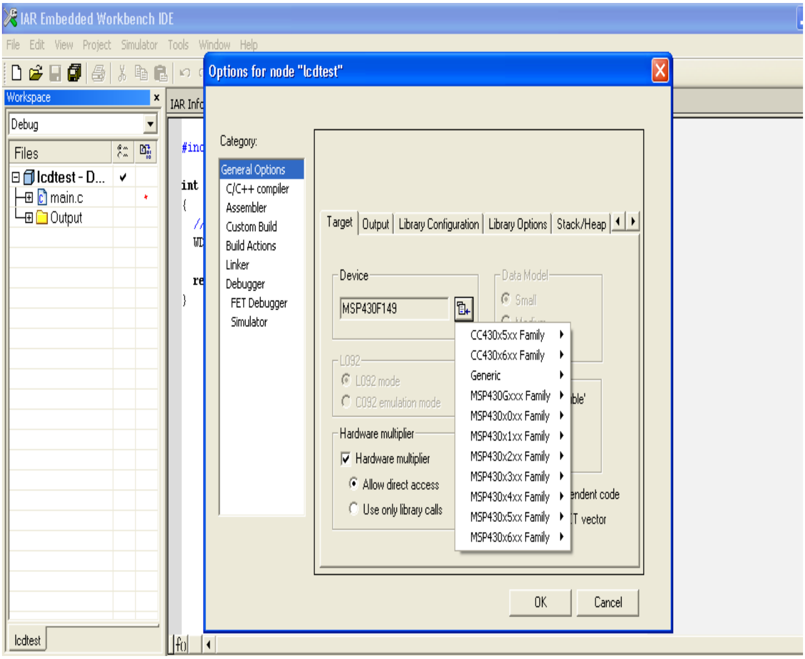
GETTING START
Guidelines for setti
your project, addin
files, compiling, lin
and debugging it.

com/resources
epth articles from our software engineer
ict managers, see video presentations a
webinar calendar or view the recordings

com/mypages
Create your user account and download product updates from MyPages. Here you can also manag your contact information and check your license a SUA status.


IAR Embedded Workbench IDE
File Edit View Project Simulator Tools Window Help
Workspace
Debug
Files
lcdtest - D...
main.c
Output
lcdtest
IAR Information Center for MSP430
main.c

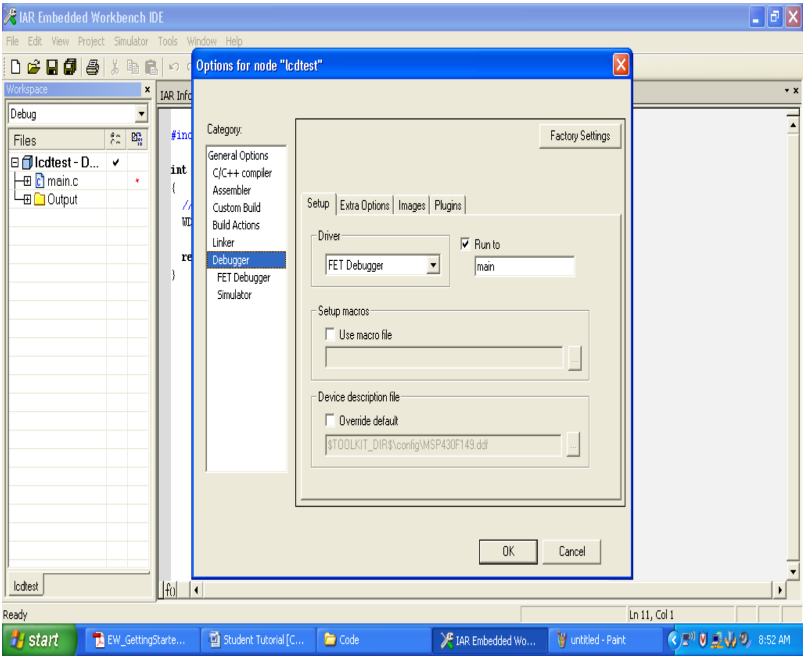
```
#include "io430.h"

int main( void )
{
  // Stop watchdog timer to prevent time out reset
  WDTCTL = WDTPW + WDTHOLD;

  return 0;
}
```


Options for node "lcdtest"

Category: General Options

Device: MSP430F149

Hardware multiplier
☑ Hardware multiplier
● Allow direct access
○ Use only library calls



Options for node "lcdtest"

Category:

- General Options
- C/C++ compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
  - FET Debugger
  - Simulator

Factory Settings

Setup | Extra Options | Images | Plugins

Driver: FET Debugger

☑ Run to: main

Setup macros

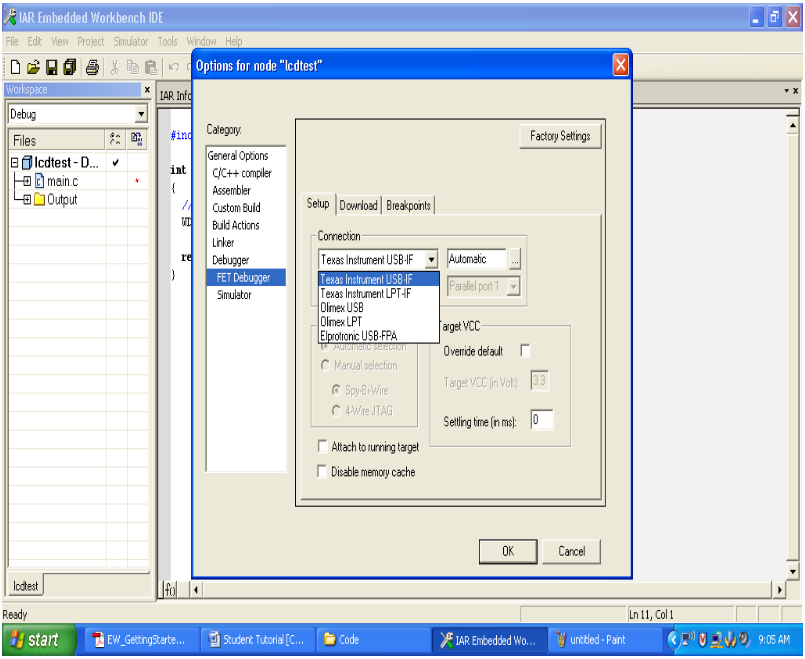
☐ Use macro file

Device description file

☐ Override default

$TOOLKIT_DIR$\config\MSP430F149.ddf

OK Cancel



Connection: Texas Instrument USB-IF | Automatic

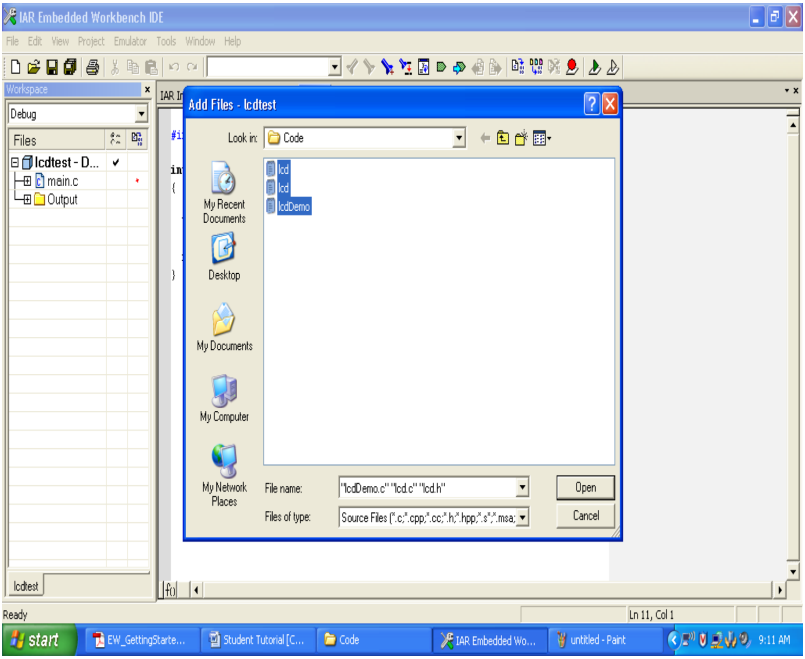
Override default ☐

Target VCC (in Volt): 3.3

Settling time (in ms): 0

☐ Attach to running target

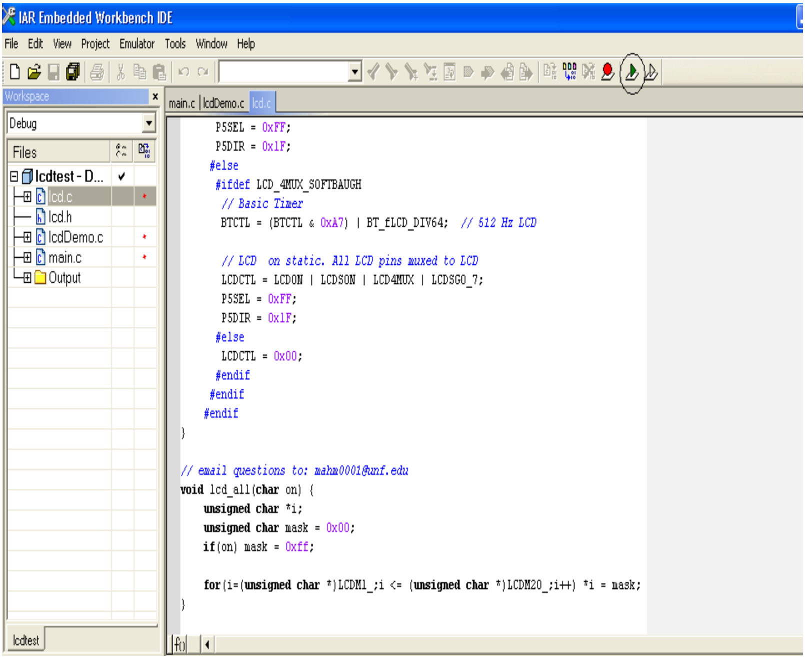
☐ Disable memory cache



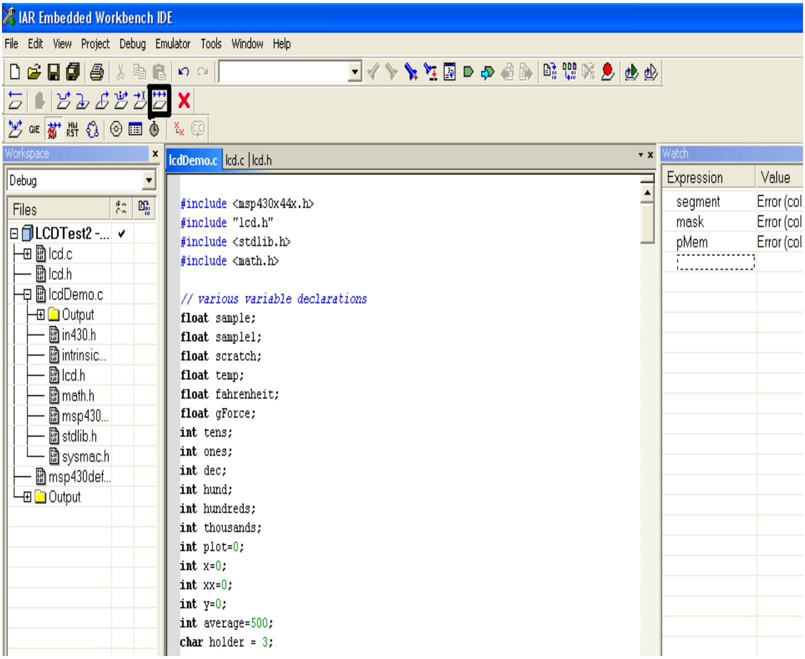
Look in: Code

File name: "lcdDemo.c" "lcd.c" "lcd.h"

Files of type: Source Files (*.c;*.cpp;*.cc;*.h;*.hpp;*.s*;*.msa;



```
      P5SEL = 0xFF;
      P5DIR = 0x1F;
     #else
      #ifdef LCD_4MUX_SOFTBAUGH
       // Basic Timer
       BTCTL = (BTCTL & 0xA7) | BT_fLCD_DIV64;  // 512 Hz LCD

       // LCD  on static. All LCD pins muxed to LCD
       LCDCTL = LCDON | LCDSON | LCD4MUX | LCDSG0_7;
       P5SEL = 0xFF;
       P5DIR = 0x1F;
      #else
       LCDCTL = 0x00;
      #endif
     #endif
    #endif
}

// email questions to: mahm0001@unf.edu
void lcd_all(char on) {
    unsigned char *i;
    unsigned char mask = 0x00;
    if(on) mask = 0xff;

    for(i=(unsigned char *)LCDM1_;i <= (unsigned char *)LCDM20_;i++) *i = mask;
}
```



```c
#include <msp430x44x.h>
#include "lcd.h"
#include <stdlib.h>
#include <math.h>

// various variable declarations
float sample;
float sample1;
float scratch;
float temp;
float fahrenheit;
float gForce;
int tens;
int ones;
int dec;
int hund;
int hundreds;
int thousands;
int plot=0;
int x=0;
int xx=0;
int y=0;
int average=500;
char holder = 3;
```

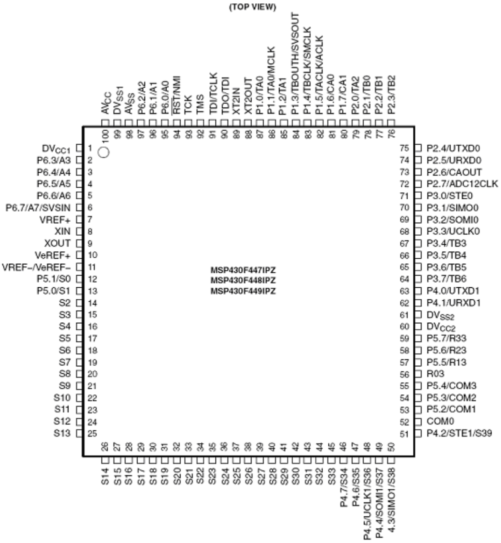
Expression	Value
segment	Error (col
mask	Error (col
pMem	Error (col

Appendix

The following specifies the pin-out connections of the MSP430F449 Texas Instrument Mixed Signal Micro controller. Also, a general outline of the physical package is included. For more data on the device, check the following web site.

http://focus.ti.com/docs/prod/folders/print/msp430f449.htmi#technicaldocuments

Look under technical documents/data sheets.



Acknowledgements:

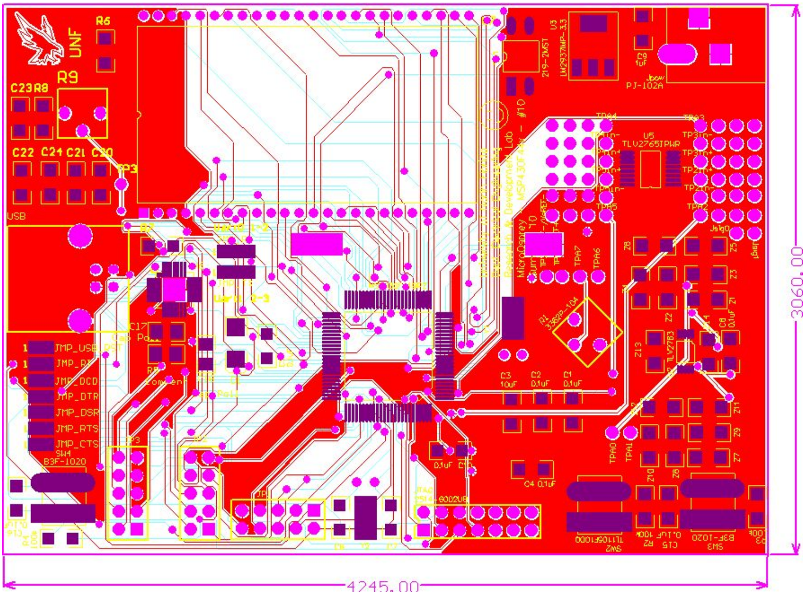
The University of North Florida (UNF) EE team began using the TI MSP430F449 a number of years ago. Bethune-Cookman University and the University of Tennessee (UT) adopted UNF’s lab kits into their curriculum. Dr. Walter Green, UT Electrical Engineering, had his students prepare a users guide for the lab. UNF students adopted the UT user’s guide and updated it per UNF current embedded system design activities. UNF thanks UT for their help and contribution to this users guide.

EEL3111 Project PCB
This is the Printed Circuit Board that is used for student projects in EEL3111 Basic Circuit Theory.

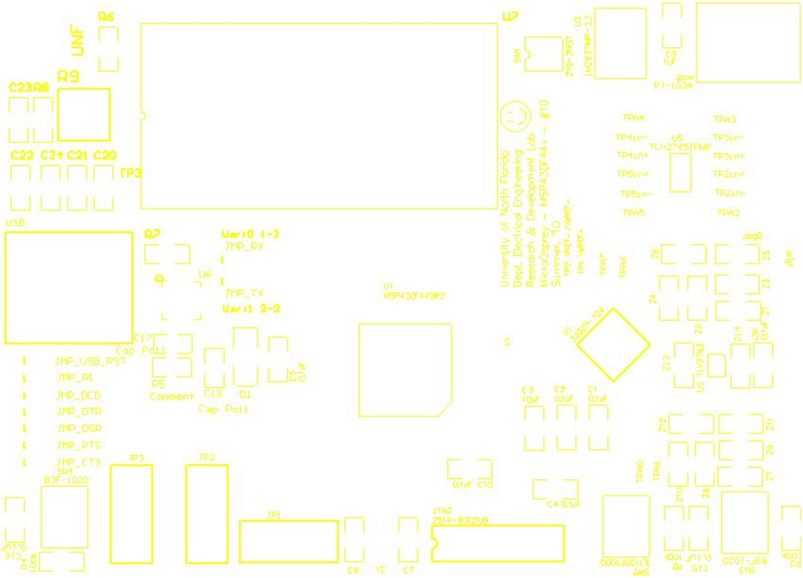
Click the following link to view a PDF version of the design files. Msp430v1.

Click the following link download a .zip folder containing the .CAM and Gerber files required for fabrication. Gerber Files.

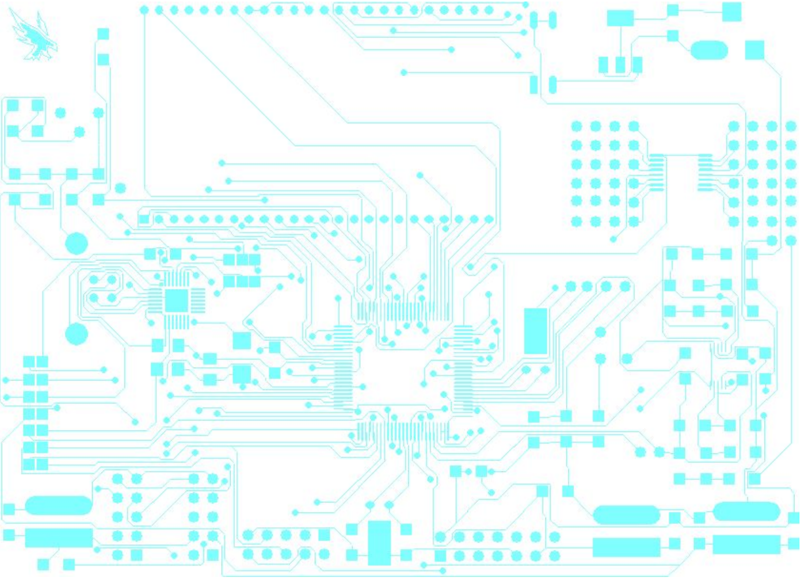
Full Composite:



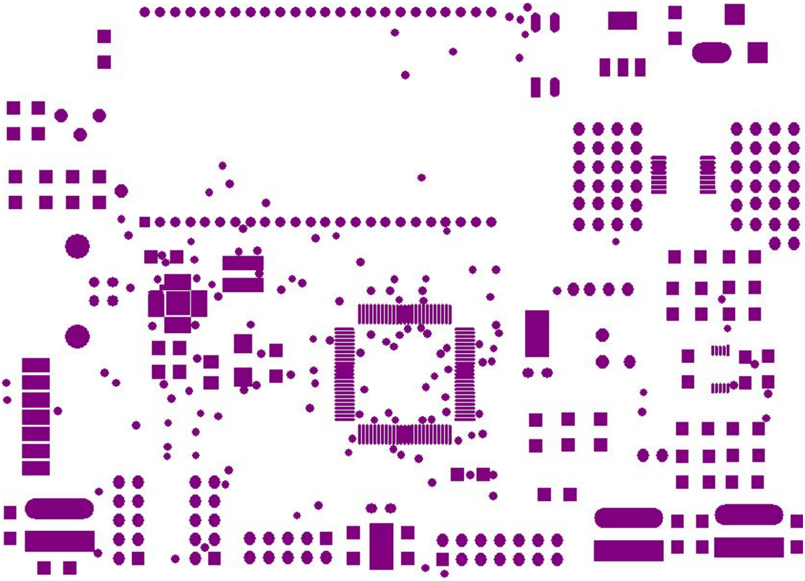
Top Overlay:



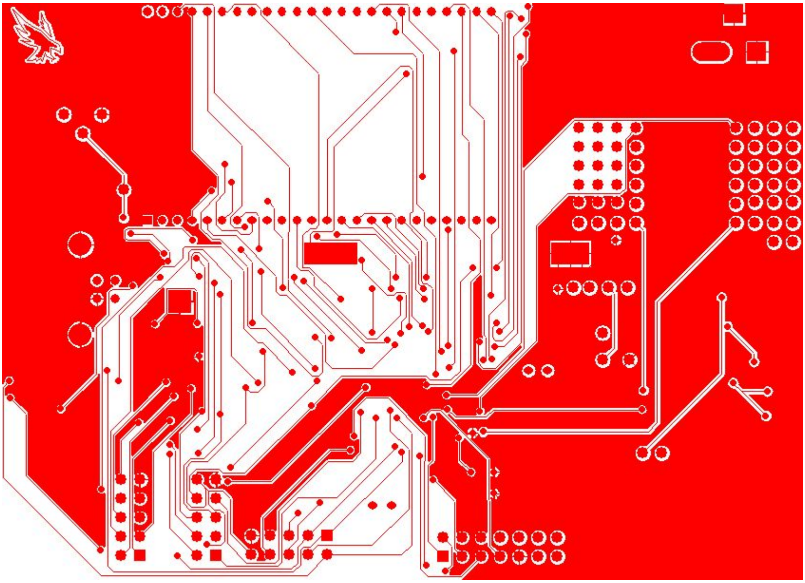
Top Copper Layer:



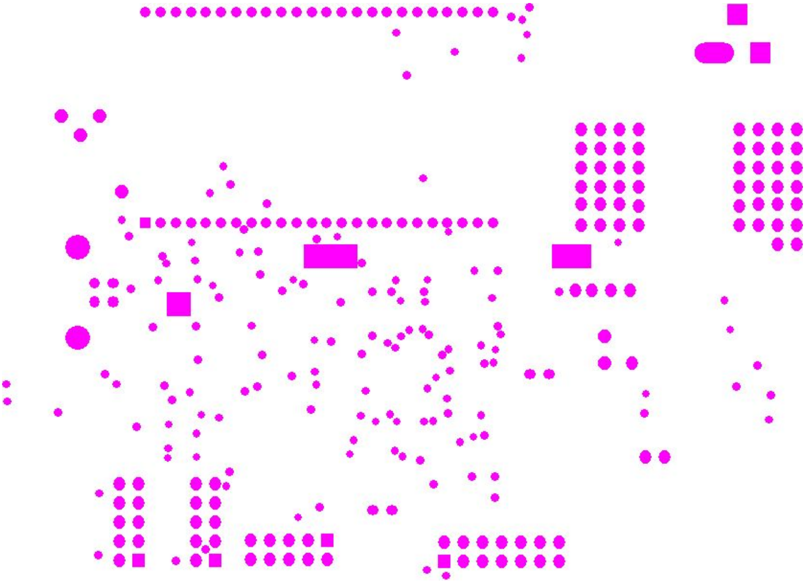
Top Solder Mask:



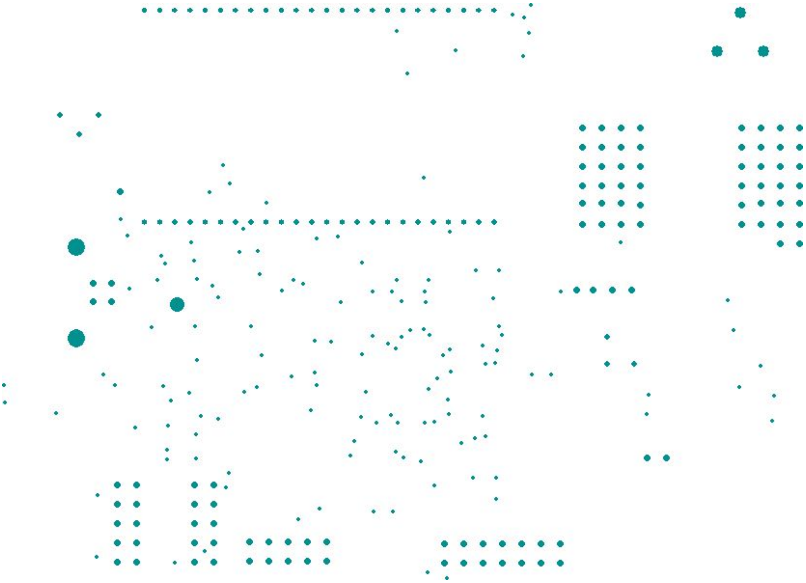
Bottom Copper Layer:



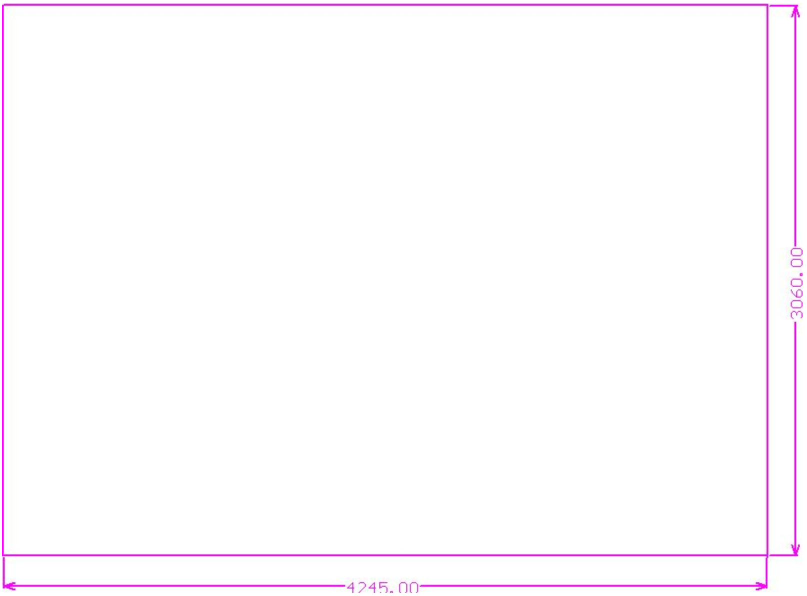
Bottom Solder Mask:



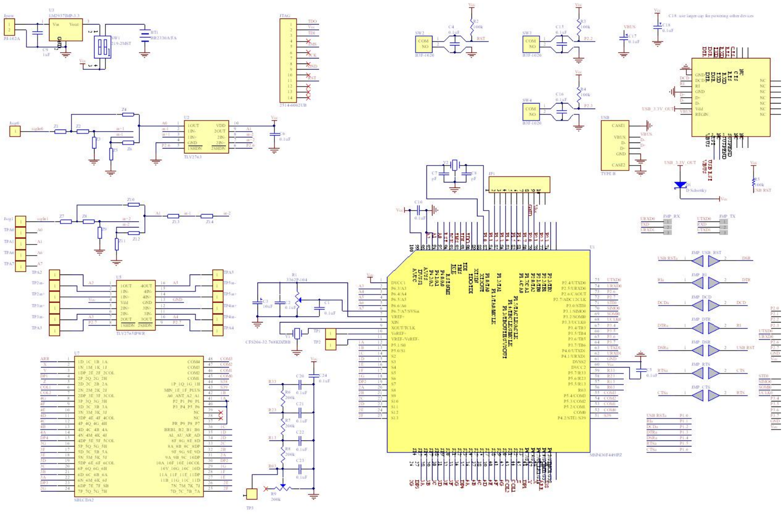
Drill Layer:



PCB Outline:



Schematic: (note clearer version in PDF)


Jpow
PJ-102A
U3
LM2937IMP-3.3
Vin
Vout
GND
C9
1uF
SW1
219-2MST
BT1
BR2330A/FA
Vcc
JTAG
TDO
TDI
TMS
TCK
GND
RST
2514-6002UB
Jsig0
sigIn0
Z1
Z2
Z3
Z4
Z5
Z6
U2
TLV2763
C6
0.1uF
Jsig1
sigIn1
Z7
Z8
Z9
Z10
Z11
Z12
Z13
Z14
TPA0
TPA1
TPA6
TPA7
TPA2
TPA3
TPA4
TPA5
U5
TLV2764IPWR
U7
SBLCDA2
R1
3362P-104
C3
10uF
C2
0.1uF
C1
0.1uF
Y1
CFS206-32.768KDZBB
TP1
TP2
TP3
R6
R7
R8
R9
200k
C20
C21
C22
C23
C24
0.1uF
SW2
SW3
SW4
COM
NO
B3F-1020
R2
R3
R4
100k
C4
C15
C16
RST
P2.2
P2.3
C17
C18
VBUS
C18: use larger cap for powering other devices
USB
TYPE B
CASE1
CASE2
D+
D-
USB_3.3V_OUT
D1
D Schottky
R5
USB RST
Y2
C7
C8
pF
JP1
C10
C5
U1
MSP430F449IPZ
JMP_RX
JMP_TX
URXD0
TXD
URXD1
UTXD0
RXD
UTXD1
JMP_USB_RST
JMP_RI
JMP_DCD
JMP_DTR
JMP_DSR
JMP_RTS
JMP_CTS
USB RSTa
RIa
DCDa
DTRa
DSRa
RTSa
CTSa
DSR
DTR
DCD
RI
USB RST
CTS
RTS

EEL3111 Code Listing
This module contains the C code for the Force Sensor example.

The following link contains all of the files necessary to compile the project using IAR Embedded Workbench. The zip file contains main.c lcd.c and lcd.h Code Listing.zip

To view the code listings for lcd.c and lcd.h please download the zip file and open them in any text editor.

```
/
*******************************************
* EEL3111 Student Project - Force
Sensor Group * Code by: Michael Toth *
Questions: totm0001@unf.edu
*******************************************
#include "msp430x44x.h" #include
"lcd.h" #include "math.h" #define
Average 4 // This value controls the
number of samples // to be taken before
an average is computed // and the LCD
is updated. #define Vref 3.3 // Don't
change this unless you know what you
are doing volatile unsigned int x=1;
long int ADC0; float ADC0Averaged;
float ADC0temp; float Voltage; float
Converted; unsigned int hundreds, tens,
ones, tenths; /
************************************ *
Function Prototypes
```

```
************************************/
void clock_config(void); void
adc12_config(void); //void
opamp_enable(int); void
update_lcd(void); float
convert_value(float); void
extract_digits(float); /
************************************ *
Main Loop
************************************/
void main(void) { clock_config(); //
Configure clock adc12_config(); //
Configure ADC12 // opamp_enable(1); //
1=enabled 0=disabled setupLCD(); //
Called from lcd.c, performs LCD setup
lcd_all(0); // clear the LCD // Enable
interrupts and go to sleep forever
_EINT(); while(1){ _NOP(); // loop
forever but still servicing interrupts
} } /
************************************ *
Clock Configuration Subroutine
************************************/
void clock_config(void){ WDTCTL = WDTPW
+ WDTHOLD; // Stop WDT FLL_CTL0 |=
XCAP10PF; // Configure load caps BTCTL
= BT_ADLY_250; // 250ms Interrupt IE2 |
= BTIE; // Enable BT interrupt } /
************************************ *
ADC12 Configuration Subroutine
************************************/
```

```
void adc12_config(void){ ADC12IE |=
0x01; // Enable ADCMEM0 interrupt
ADC12CTL1 = ADC12SSEL_2 | SHP; //
ADCCLK = mCLK, pulse mode ADC12MCTL0 =
SREF_0 | INCH_0; // Vr+=AVcc and Vr-
=AVss // warm-up the ADC ADC12CTL0 =
REFON | ADC12ON; // turn on ref and ADC
ADC12CTL0 |= ENC; // enable conversions
} /************************************
* Opamp Enable Subroutine: * The
project board includes a place for * a
surface mounted opamp. The opamp has *
a pin that must be set high for the
opamp * to function. The pin is
connected to P2.7 * by setting P2.7
high the opamp is enabled, * low the
opamp is disabled
************************************/ /
* void opamp_enable(int x){ P2DIR |=
BIT7; // Set P2.7 to output direction
if(x==1){ P2OUT |= BIT7; // set P2.7
high. Enables Opamp } */ /
**************************************
* Update LCD Subroutine
**************************************/
void update_lcd(void){ lcd_char(0,ones
+ '0'); lcd_char(1,tens + '0');
lcd_char(2,hundreds + '0'); } /
**************************************
* Convert Value Subroutine: * Convert
from voltage to sensor value. *
```

```
Formulas usually available in datasheet
* for specific sensors.
***************************************/
float convert_value(float voltage){
float value; value =
14.609*pow(voltage,4)-38.65*pow(voltage,3)+4
return value; } /
***************************************
* Extract Digits Subroutine: * The LCD
driver requires that each digit * be
presented to the LCD individually. i.e.
* if you want to display 3.14 on the
LCD 3, 1, * and 4 must be held in
seperate variables. * This algorithm
assumes only positive values.... *
xxx.xx can be represented
****************************************/
void extract_digits(float converted){
hundreds = (int)converted / 100; //
extract hundreds digit tens = (int)
((converted - (hundreds*100)) / 10); //
extract tens digit ones =
(int)converted - hundreds*100-(tens *
10); // extract ones digit } /
************************************ *
Interrupt Subroutines
************************************/
// Basic Timer interrupt service
routine #pragma
vector=BASICTIMER_VECTOR __interrupt
void basic_timer(void) { // start a
conversion // when conversion is
complete ADC12 interrupt will take
place ADC12CTL0 |= ADC12SC; } // ADC12
Interrupt Service Routine #pragma
vector=ADC12_VECTOR __interrupt void
ADC12ISR (void) { ADC0 = ADC12MEM0; //
IFG is cleared ADC0temp+=ADC0; //
accumulate values to average
if(x==Average){ // test if enough
values have been collected
ADC0Averaged=ADC0temp/Average; //
compute the average value
Voltage=ADC0Averaged*Vref; // multiply
by Vref to obtain voltage representaion
Voltage=Voltage/4095; // divide by
#quantization levels
Converted=convert_value(Voltage); //
Convert voltage to appropriate units
extract_digits(Converted); // Extract
individual digits update_lcd(); //
Update the LCD display x=1; // Reset
the averaging loop ADC0temp=0; } else{
x++; } }
```